Passive Identification of IoT Devices from IEEE 802.11 Frames Using Machine Learning

University of Turku
Department of Computing
Master of Science (Tech) Thesis
Cyber Security Engineering
May 2026
Valtter Karppinen

Supervisors:
Saku Lindroos
Antti Hakkala

UNIVERSITY OF TURKU
Department of Computing

Valtter Karppinen: Passive Identification of IoT Devices from IEEE 802.11 Frames Using Machine Learning

Master of Science (Tech) Thesis, 73 p., 1 app. p.
Cyber Security Engineering
May 2026

This thesis investigates the identification of internet of things devices in wireless local area networks based on individual IEEE 802.11 frames using passive monitoring. The primary methodology utilizes machine learning, specifically a dual-branch fusion architecture combining a one-dimensional convolutional neural network and a multilayer perceptron. The classification model evaluates network traffic using an input consisting of raw data formed by the first 49 bytes of individual IEEE 802.11 frames and metadata describing traffic dynamics, such as frame inter-arrival times and total lengths. The model's actual generalization capability to completely new devices was tested through device-aware cross-validation and probability calibration. This method utilized network traffic data combined from multiple open-access network traffic sources.

The results indicate that the developed fusion model is capable of identifying the majority of IoT device network traffic while maintaining a low number of false alarms on a test set. Comparison with ablation models confirmed that reliable identification benefits from combining both structural protocol information and traffic dynamics. Although the method provides a computationally efficient tool for real-time network access management, the standardized structure of the IEEE 802.11 protocol establishes a natural upper limit on the identification accuracy of individual frames. In the future, the system's reliability in real world environments could be improved by integrating semi-supervised or active learning and by utilizing increasingly diverse training data.

Keywords: Internet of Things, device identification, IEEE 802.11, machine learning, WLAN, cybersecurity, network traffic analysis

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background and Significance

The increasing prevalence of internet of things (IoT) devices in wireless networks has made device identification a critical aspect of managing and securing wireless local area network (WLAN) traffic. The exponential growth in the number of IoT devices has resulted in ubiquitous device presence across diverse environments, such as industrial systems, smart homes, and personal healthcare systems. Because IoT devices are generally equipped with resource-constrained hardware, a significant number of devices lack adequate security measures, rendering the devices susceptible to numerous network threats, privacy issues, data integrity problems, and confidentiality vulnerabilities [1] [2]. Consequently, IoT devices are often the most vulnerable components of both business systems and consumer settings [1]. The inherent hardware and software vulnerability makes companies and consumers alike more susceptible to cyberattacks [1].

Unidentified or unauthorized IoT devices can introduce significant security vulnerabilities, often becoming targets or participants in malicious activities such as botnet attacks. In smart home environments, the primary security threats include eavesdropping, distributed denial-of-service (DDoS) attacks, and unauthorized access to information. For example, compromised smart locks can grant attackers unauthorized physical access to the premises [1]. Furthermore, poorly secured de-

vices may be co-opted into DDoS botnets without the user's knowledge [1]. A prominent example of compromised IoT devices is the Mirai botnet, which managed to co-opt nearly half a million devices, including routers, digital video recorders, and security cameras [3].

Accurate identification of IoT devices provides network visibility for administrators to mitigate inherent security risks. When the device type is reliably known, operators can implement targeted security measures, such as network segmentation, which isolates vulnerable IoT devices into restricted zones and prevents potential malware from spreading across the infrastructure [4]. Furthermore, device-specific identification enables efficient resource allocation by allowing administrators to provision appropriate quality of service policies based on the predictable traffic requirements of each device category. Consequently, continuous and automated device profiling is a necessary requirement for maintaining both the operational reliability and the security posture of modern wireless networks.

Despite the growing importance of IoT device identification, existing methods often focus on active network analysis, which relies on active network connections and partially controlled network parameters [5] [4] [6] [7]. Furthermore, because the application-layer payload of modern network traffic is typically encrypted, device identification using traditional content inspection methods is ineffective [8]. Device identification must therefore rely on unencrypted characteristics of network traffic, such as data link layer header information [8]. The research presented in this thesis investigates how IoT device identification succeeds in passive network analysis when dealing with unidentified devices and networks. The objective is to distinguish frames sent by IoT devices from other network traffic at the medium access control (MAC) layer, thereby creating a technical foundation for the automation of device management and access control.

1.2 Problem statement

Reliable identification of IoT devices within real-world WLAN traffic remains challenging, especially when conducting passive analysis at the level of individual frames defined by the IEEE 802.11 WLAN standard. Current identification methods predominantly rely on controlled laboratory environments, where IoT devices are directly connected to gateways or dedicated capture devices. Such controlled data lacks the natural variability present in real network environments. However, in real urban wireless networks the traffic is dynamic, intermittent, and unlabeled, which makes it difficult to identify IoT devices. The effectiveness of existing frame-level IoT identification techniques is limited under these realistic conditions, as IEEE 802.11 frames frequently include noise, varied device interactions, and temporarily visible devices.

This thesis develops a machine learning method aimed at identifying IoT devices from IEEE 802.11 WLAN traffic data captured passively at the frame level using monitor mode. Unlike captures performed in standard managed mode, monitor mode allows detection of all IEEE 802.11 WLAN frames, including management and control frames, beacon frames, and probe requests, even from devices not associated with an access point (AP). This level of detail is essential for identifying IoT devices based solely on observable wireless behavior. The motivation for this research is the growing need for reliable IoT device identification to enhance network manageability and security. Since IoT devices are often the weakest link in network security, their precise identification is important for preventing cybersecurity threats and unauthorized network access.

The study addresses the following research questions:

1. How reliably can IoT devices be identified from dynamic IEEE 802.11 WLAN traffic using machine learning models based on individual MAC frames captured passively in monitor mode?

2. What machine learning architecture is most suitable for processing structural and dynamic single-frame features for IoT device identification, and what level of classification performance can be achieved?

By answering these questions, the thesis contributes to the field of passive IoT device identification and establishes practical foundations for improving network security in realistic urban WLAN environments.

1.3 Structure of the thesis

Chapter 2 provides a theoretical framework for the study by examining the network behavior of IoT devices, the structure of the IEEE 802.11 WLAN standard, and previous research on device identification based on machine learning. The chapter discusses in more detail how the protocol-level characteristics of such WLAN traffic appear in passive observation, and why methods relying solely on payload content are insufficient in modern encrypted network environments. In addition, the chapter provides background on the role of deep learning in representation learning from network traffic data.

Chapter 3 presents the experimental methodology of the study, covering network traffic data collection, data preprocessing, and the selected neural network architecture and evaluation protocol. The chapter describes how raw frames are converted into one-dimensional sequences and how dynamic features of network traffic are combined with these to form a unified input. At the same time, the choice of a two-branch hybrid architecture is justified, and the principles of device-level cross-validation are defined, aiming to ensure the model's actual ability to identify new devices.

The technical implementation of the developed classification models and the evaluation of experiment results are discussed in Chapter 4. The chapter compares

the performance of the fusion architecture with two simplified ablation models using a test set. In addition to numerical classification metrics, the results are analyzed from the perspective of the model's robustness and feature shift. The decision-making logic is interpreted using saliency maps and two-dimensional feature space projections.

Chapter 5 evaluates the method's suitability, limitations, and needs for further development in the context of cybersecurity. The chapter discusses how the achieved results are applicable to operational use. In addition, it examines the structural and protocol-level limitations associated with the analysis of a single frame, which form a natural upper bound on recognition accuracy, and presents recommendations for future work. Finally, Chapter 6 summarizes the study's key findings and concludes the thesis.

2 Theory and literature review

2.1 IoT devices in Wireless Networks

The operation of IoT devices in WLANs introduces structural and behavioral characteristics unique to network traffic. This section reviews the physical characteristics of these devices and the communication protocols that allow passive identification of devices in wireless environments, as well as the security challenges that make this identification necessary.

2.1.1 Types of IoT Devices

IoT is not a single technology or device but a combination of technologies, devices, and functions that create the IoT paradigm [9]. IoT devices differ from traditional devices in several ways and can be identified by their distinct characteristics. The physical attributes, communication protocols, networks used, data transfer techniques, and operating environments of IoT devices significantly differ from those of traditional devices. For example, IoT devices often use lightweight protocols and are designed to operate in different environments compared to traditional devices [9]. Additionally, data transfer in IoT devices is often optimized for low power consumption and limited bandwidth [9].

In terms of physical characteristics, IoT devices can vary greatly in appearance and size. However, an IoT device typically includes [9]:

Sensors: Components that perceive environmental conditions such as temperature, humidity, or motion.

Processors: Units that handle data processing tasks.

Actuators: Mechanisms that can manipulate physical systems in response to data inputs.

Communication Modules: Capabilities to send and receive stored or processed data.

IoT devices usually have constraints due to their application, cost, or other reasons. These constraints may include limited computing power, small storage capacity, size restrictions, or energy consumption limitations [9].

Examples of IoT devices used by consumers include a robotic vacuum cleaner, a smart thermostat or a smartwatch. These devices are often connected to the network using a wireless technology such as WLAN or Bluetooth [10]. These devices often lack comprehensive user interfaces and are managed via mobile devices or computers. IoT devices operate on the same principle in an industrial environment, but the requirements and users differ [10]. Industrial IoT devices have higher requirements than consumer IoT devices. In industry, IoT devices need to be reliable, capable of collecting data, used on a larger scale and the operating environment can often be challenging [10].

It is widely acknowledged that IoT devices present a significant security risk. However, industrial IoT devices are particularly vulnerable due to their deployment in high-risk environments where they directly or indirectly impact production and the economy [11]. Consequently, they are more susceptible to cyber threats than consumer IoT devices [11].

Due to the high security threat posed by IoT devices, it is important to develop methods for identifying these devices within their operational environment. This thesis studies ways to passively identify IoT devices from the surrounding network

traffic in order to identify unidentified IoT devices and mitigate potential threats to these devices.

2.1.2 Communication Technologies and Protocols Used by IoT devices

Communication protocols are essential for the seamless operation of IoT devices, facilitating efficient data exchange and ensuring interoperability across heterogeneous networks [9]. In the context of passive WLAN traffic analysis, IoT devices can be broadly categorized based on their network connectivity architecture. Some devices, such as smart plugs or Internet Protocol (IP) cameras, implement a full Transmission Control Protocol/Internet Protocol (TCP/IP) stack and connect directly to the WLAN network [12]. Conversely, many resource-constrained sensors and actuators utilize specialized low-power communication technologies, such as Zigbee or Bluetooth Low Energy (BLE), which do not inherently operate over IP networks [7]. These devices require a dedicated gateway or hub to translate their proprietary or low-power protocols into standard IP traffic before it can be transmitted over WLAN connection [7].

This architectural distinction impacts the process of device identification through passive network monitoring. When analyzing IEEE 802.11 WLAN frames passively, a system only observes frame information of the direct network participant. Consequently, since the underlying physical sensors utilize alternative communication protocols, they do not operate as direct IEEE 802.11 WLAN clients, and the passive analysis identifies only the intermediary gateway [7]. Multiple distinct end-devices connecting through a single intermediary gateway manifest as a single logical entity in the WLAN traffic, masking the specific hardware characteristics and individual transmission rhythms of the original data sources [7].

For devices that do communicate across the IP network, the application-layer protocols further define their traffic profiles. Protocols such as Message Queuing Telemetry Transport (MQTT) and Constrained Application Protocol (CoAP) are widely deployed to accommodate the limited processing power and bandwidth constraints typical of IoT environments. While MQTT provides a lightweight publish-subscribe messaging transport optimized for unreliable networks, CoAP offers a RESTful architectural style adapted for low-power communication [9]. Although the payloads of these protocols are typically encrypted in modern deployments, their structural behaviors, connection persistence, and communication frequencies contribute to the observable traffic patterns utilized in passive device fingerprinting [4] [7].

2.1.3 The impact of IoT on WLAN traffic

The introduction of IoT devices into wireless networks has introduced new patterns in data transmission. IoT devices typically generate small-sized data packets sent at regular intervals, leading to frequent, low-volume transmissions that influence WLAN traffic characteristics. This type of transmission often appears as periodic updates, where devices regularly transmit sensor readings or status updates to a central gateway. Such consistent and frequent data exchanges differ from traditional internet usage patterns, which are generally bursty and involve larger data packets during shorter periods of activity, such as browsing or streaming video content. [12]

In contrast to IoT devices, traditional client devices display distinct behaviors in terms of traffic generation and data flow. Devices such as laptops and smartphones typically produce bursty, interactive data traffic characterized by sporadic high-volume transmissions that correspond with user-driven actions. For instance, web browsing or media streaming sessions generate substantial traffic intermittently, resulting in distinct peaks and troughs in traffic volume. Unlike the continuous and

predictable data patterns from IoT devices, these bursts cause varying loads on network resources, which may require adaptive traffic management approaches. [13]

These differing transmission patterns can influence overall WLAN performance, particularly under dense deployment scenarios. Frequent small-packet transmissions from IoT devices can cumulatively lead to network congestion, as WLAN protocols must handle increased contention and transmission overhead. The resulting contention can degrade overall network performance, leading to increased latency and reduced throughput. Therefore, identifying IoT devices is an important factor in network management to optimize network performance. By effectively managing all devices on the network, the quality of service for all devices connected to the network can be guaranteed. [14]

2.1.4 Current challenges of IoT security

The security of IoT devices is often compromised by fundamental vulnerabilities inherent in their design and deployment [15] [16]. Like any other network-connected device, IoT devices are vulnerable to traditional security threats such as data tampering, unauthorized access to the device, or leaking sensitive information [15]. However, due to their inherent resource constraints and specialized architectures, IoT devices operate differently compared to traditional WLAN client devices, exposing them to other security threats as well [15] [16]. These threats can be broadly categorized according to the general IoT architecture: physical device threats, network threats, and application threats [15]. These vulnerabilities expose devices, and therefore the entire network, to threats that affect the information being processed and transmitted, network access control and management, and functionality [16].

The physical device is the base of the IoT, through which data is always transmitted. This includes the physical properties of the device, such as radio frequency identification (RFID), sensors or a possible battery. Even if the device's connectiv-

ity, data or applications cannot be affected, the device may be vulnerable to physical threats. Such physical threats to a device include, for example, the corruption of sensors or the replacement of a device with a new contaminated device. [15] Such actions may not allow direct access to the network or data, but may instead be disruptive.

Network threats mainly concern the protocols used for communication and vulnerabilities in the design of IoT networks. To accommodate resource constraints, IoT devices utilize lightweight application-layer protocols, such as CoAP and MQTT, alongside specialized low-power communication technologies, such as Zigbee [15] [16]. Because these protocols and technologies are designed to be as lightweight and energy-efficient as possible, they often have less robust security mechanisms and weaker encryption technologies than other protocols [16]. These insufficiencies leave traffic vulnerable to eavesdropping or unauthorised manipulation, for example [16].

Application threats attempt to exploit vulnerabilities in software. Software vulnerabilities are relatively easy to exploit, as IoT devices may often have software that is not properly maintained. In the case of unmaintained software, the processes of the devices can be directly manipulated [16]. Poor software also exposes devices to authentication and authorisation problems due to weak access management or default passwords [15].

Some threats can affect more than one layer of the IoT architecture, such as firmware vulnerabilities. Their exploitation in attacks has increased recently. Many IoT devices already have poorly tested and configured firmware, or are otherwise weakened by constraints. Furthermore, some devices are equipped with firmware that cannot be updated, which makes it impossible to apply patches for newly discovered vulnerabilities [17]. Consequently, adversaries exploit these inherent flaws either to disrupt standard device operations or to utilize the compromised node as an initial entry point for broader network attacks [16].

The large number of vulnerabilities in IoT devices and networks and their poor maintenance will increase the security debt [18]. With the growth in the number of IoT devices and the overall poor security posture of IoT, the security debt can be expected to increase significantly in the future. The widespread deployment of IoT devices has been compared to the historical use of asbestos, as their integration introduces severe but latent security vulnerabilities [19]. Many devices are being connected on the internet at a fast pace as an easy solution, but the large number of IoT devices could be even more challenging in the future when the risks start to realise.

2.2 Overview of the IEEE 802.11 WLAN standard

2.2.1 802.11 Standards

This thesis focuses on the network traffic of the IEEE 802.11 standard. 802.11 is a commonly used standard for WLAN technology that defines the operation and protocols of the medium access control and physical layers for WLANs [20]. Since the original version, several amendments have been introduced as part of the IEEE 802.11 standard to improve data transfer rates, spectral efficiency, and channelization [21] [20].

Each amendment of the standard includes upgrades and improvements over the previous version, such as increased data rate, better energy efficiency and improved channelisation. For example, the 802.11b standard introduced complementary code keying (CCK) coding technology, which allowed data rates to increase from 2 Mbps to 11 Mbps, while the 802.11a standard introduced orthogonal frequency-division multiplexing (OFDM) technology, which is also used in current mobile network technologies such as 5G [22]. This further increased speeds to a theoretical rate of 54 Mbps. Subsequent updates to the standard, such as 802.11n, introduced multiple-

input and multiple-output (MIMO) technology, which allowed the theoretical maximum speed to be increased to 600 Mbps. Even the most recent standards, such as 802.11ax and 802.11be, continue to make use of these technologies. The latest updates to the standard have also introduced new frequency bands. The most common frequencies in use today are 2.4 GHz, 5 GHz and, most recently, 6 GHz. [20]

Updates to standards have brought new solutions to improve security. For example, Wi-Fi Protected Access 3 (WPA3), released in 2018, significantly improved the security of WLAN networks. It uses the Simultaneous Authentication of Equals (SAE) authentication protocol to replace the pre-shared key (PSK) protocol of the previous Wi-Fi Protected Access 2 (WPA2) security standard. SAE enables stronger encryption, and makes it impossible to carry out, for example, offline attacks in the WPA3 security standard. [23] WPA3 uses the forward secrecy feature, which ensures that even if the private key used for encryption is broken or stolen, it cannot be used to decrypt previously intercepted data [24]. Despite the improvements, WPA3 has also been shown to be vulnerable to attacks. Research shows that WPA3 is vulnerable to downgrade attacks, where a device is forced to use a previous, vulnerable security protocol in the name of compatibility [25].

The evolution of WLAN networks has also contributed to the development of IoT. Improvements in the 802.11 standard, such as higher data rates, better support for the use of different frequency bands and improved energy efficiency, have enabled the proliferation of IoT devices in homes and industrial environments. For example, the 802.11ah standard, known as Wi-Fi HaLow, is specifically designed for IoT devices. It offers longer range, low power consumption, the ability to support thousands of devices simultaneously [26]. These are all essential for modern IoT networks. In addition, the latest standard amendments, such as 802.11ax, enable denser IoT networks by reducing channel congestion and improving spectrum efficiency [26] [27]. These 802.11 standard amendments have enabled an increase in the number of IoT

devices in IoT networks and improved their performance.

2.2.2 IEEE 802.11 Frame Structures

WLAN communications are based on the frameworks defined by the IEEE 802.11 standard, which govern the transmission of data over a wireless network. In this work, we consider traffic records composed of these frames, which contain both packet-level and frame-level information. The goal of the analysis is to utilize this data to identify IoT devices.

An IEEE 802.11 MAC frame comprises a set of three fields: a MAC header, a frame body, and a frame check sequence (FCS) [20]. Frames can be categorized into three primary types: management, control, and data frames [21]. Management frames facilitate the establishment and maintenance of network communication between client devices and APs, including processes such as authentication, association, and beaconing. Control frames coordinate access to the wireless medium and support data transmission, using subtypes such as request to send (RTS), clear to send (CTS), and acknowledgement (ACK). Data frames serve as carriers for the actual user payload, which may encapsulate multiple higher-layer protocols within their bodies.

The structure of each data frame includes several components: the MAC header, the frame body, and the FCS. The MAC header which is typically 30 bytes in length, includes addressing and control information. It contains fields such as Frame Control, Duration, multiple address fields (source, destination, and transmitting device), Sequence Control, and in some cases Quality of Service control information. The frame body varies in size depending on the frame type and the payload carried, reaching up to 2312 bytes. The FCS, consisting of 4 bytes, is appended for error detection purposes, applied to both the MAC header and frame body. [28]

2.2.3 Common Traffic Patterns

WLAN traffic is inherently variable, as it is affected by the devices, applications, and software configurations in use. Such variation may occur in client session durations, packet sizes, and traffic intensity [29].

Devices used and controlled by humans, such as computers and mobile phones, are those that generate highly variable traffic. Such traffic is mainly generated by user activity and complex activites, which results in intermittent and irregular requests to connect to network services. In an extensive WLAN network analysis, Alipour et al. [30] found that user traffic is distributed in a highly variable temporal and spatial manner. Network connections often consist of short but variable data sessions, the rhythm of which is determined by user activity.

IoT devices differ from user-controlled devices in their traffic behaviour. They typically generate regular and predictable traffic based on predefined activities and schedules [4]. Chowdhury et al. [31] analyzed traffic patterns of IoT devices at both the packet and frame level and found that many IoT devices use standard traffic characteristics such as repeated requests to the same servers, a limited number of ports, and predefined delays between packets. Such protocol-level regularity is in stark contrast to user devices, where network traffic varies dynamically according to active usage [4]. Chowdhury et al. demonstrated that exploiting unique traffic patterns enables the network identification of IoT devices, which is essential for both network security and resource management [31].

2.2.4 Impact of IEEE 802.11 Security Mechanisms on Traffic Visibility

Modern IEEE 802.11 networks utilize security standards such as WPA2 and WPA3 to ensure the confidentiality and integrity of wireless communications [25]. The in-

cluded security mechanisms encrypt the application-level payload of data frames, rendering traditional content-based analysis methods ineffective [8]. Although encryption protects the payload from third parties, it does not conceal all network traffic metadata, enabling the passive profiling of devices [8].

Basic network functions, such as forwarding traffic to the correct addresses and managing radio channel reservations, require link-level metadata to be readable by all network devices. Consequently, MAC layer header information is inherently transmitted unencrypted on the network [32]. Payload encryption also does not cover external characteristics of network traffic, such as total frame length or inter-arrival times (IATs) [8]. Visible protocol-level metadata and device transmission dynamics form structural features. Passive network traffic analysis and machine learning models can utilize unencrypted characteristics to identify devices without the need to decrypt the actual payload [32] [8].

2.3 Identifying Characteristics of IoT Devices

2.3.1 Key Characteristics of 802.11 Frames

In the analysis of individual packets, a primary distinguishing factor is packet length [8]. Even if the network traffic is encrypted, the packet size remains visible and reveals information about the operation or the protocol of the device [32]. Studies have shown that packets sent by IoT devices typically follow standard lengths that differ from the dynamic traffic generated by traditional computers and smartphones [8]. For instance, certain sensors or smart home devices regularly send status updates or handshake messages of a specific size, which creates a distinct traffic profile for the device [7] [4].

Aksoy and Gunes [33] demonstrated the possibility of identifying IoT devices based solely on individual packets, eliminating the need to monitor extended traf-

fic flows. Their SysID system established that a single TCP/IP packet contains sufficient information for reliable device classification. The study utilized genetic algorithms to select optimal features from packet headers, achieving an average classification accuracy of 82% across 23 distinct IoT devices. The achieved accuracy confirms that automated device identification is viable without domain-expert input or long-term statistical monitoring. However, the SysID system focuses on distinguishing between specific IoT devices, whereas the objective of this thesis is to perform binary classification separating IoT traffic from non-IoT traffic.

Single-packet device identification exploits the fact that distinct device manufacturers and software implementations process network protocols differently, leaving unique structural traces in the transmitted packets [33]. Aksoy and Gunes [33] demonstrated that the combination of packet header fields, such as window sizes and TCP flags, and overall packet length acts as a robust device fingerprint without relying on explicit network identifiers. Single-packet analysis is particularly advantageous in environments where IoT devices transmit data infrequently, rendering traditional time-series analysis of continuous traffic flows impractical.

In addition to length and header fields, the distribution and values of the bytes in a packet form a statistical fingerprint. When the contents of a packet are treated as a whole, for example by converting the bytes of the packet into numerical values or an image matrix, structural features can be detected that are not related solely to individual protocol fields but to the overall structure of the packet [34] [35]. These type of representations make it possible to detect subtle differences that are not apparent in traditional metadata analysis [34] [36].

2.3.2 Header Information And Protocol-Based Features

Although much of modern network traffic, including communication between IoT devices, is encrypted (e.g., with TLS/SSL protocols), header information is often

transmitted in plain text to enable routing and connection management [37]. These headers contain protocol-specific parameters that can reveal the identity of the IoT device or characteristics of the firmware the device uses. Studies have shown that the header fields of the TCP/IP and MAC layers alone can be sufficient for classifying devices with high accuracy without the need to analyze the actual payload of the packet [31] [33].

At the transport layer, TCP and User Datagram Protocol (UDP) headers provide significant distinguishing features. In particular, the TCP window size and its scaling parameters are important identifiers, as they vary between different operating systems and network stack implementations [33]. In addition, the source and destination ports used form device-specific profiles. Sivanathan et al. [4] observed that, unlike general-purpose computers, IoT devices often communicate with a limited set of devices using specific ports. For example, certain devices consistently use the Network Time Protocol (NTP) port 123 for time synchronization or the Domain Name System (DNS) port 53 for name services, while others may use manufacturer-specific ports for command and control traffic. The behavior of TCP sequence numbers and packet retransmission mechanisms can also serve as device-specific characteristics [31].

At the network layer, IP header fields such as time-to-live (TTL) provide clues about the underlying operating system of the device, as different operating systems use different default values for the TTL field [33]. At the link layer, management frames compliant with the IEEE 802.11 standard, such as probe request messages, are useful for identification. Chowdhury et al. [31] showed that MAC-level features, such as FCS and sequence numbers, vary between device manufacturers and can help distinguish between different device models from the same manufacturer.

In addition, the handshake phases of encrypted traffic leak information about the device. Even if the application data is encrypted, the "Client Hello" message

sent during TLS connection establishment contains a list of encryption algorithms (Cipher Suites) supported by the device. This list often remains constant from one session to another. [4] This protocol-based metadata, together with the structural characteristics of the packets, forms a basis that can be used as input in machine learning models when traffic is processed as raw data or as an image representation.

2.3.3 Representation of 802.11 Frames for Deep Learning

The adoption of deep learning in network traffic analysis has enabled a shift from traditional feature engineering to representation learning [38]. In this approach, raw network traffic data is fed into the model in a format that preserves the original structure and information of the data. Direct utilization of raw frame bytes allows convolutional neural networks (CNNs) to automatically extract structural regularities from the traffic [34] [35].

Previous research has frequently converted network packets into two-dimensional visual representations to leverage standard image classification techniques. Since the bytes in a single network packet range from 0 to 255, they can be directly mapped to pixel intensity values [38]. When consecutive bytes are arranged in a two-dimensional matrix, protocol structures form visual patterns. For example, Bendiab et al. [34] and Hu et al. [38] converted network traffic into two-dimensional images and utilized two-dimensional CNN for traffic classification. Similarly, Hartpence [35] demonstrated that converting packets into images allows CNN models to recognize protocol-specific features without predefining relevant header fields.

While the two-dimensional representation exploits spatial correlations common in computer vision, it does not align with the inherent structure of network protocols. Network traffic is fundamentally one-dimensional sequential data, and forcing raw packet bytes into a two-dimensional grid creates artificial spatial adjacencies between bytes that possess no logical relationship in the protocol stack. To avoid

these structural distortions, this thesis utilizes one-dimensional convolutional layers. The one-dimensional convolutional neural network (1D-CNN) architecture processes traffic as a natural sequence, which preserves the true order of protocol fields and ensures that the convolutional filters react directly to the dependencies formed by consecutive bytes. Due to the rigid and repetitive structure of IoT network traffic, this sequential analysis provides a foundation for extracting device-specific fingerprints without the need for two-dimensional transformations, as previously demonstrated by Liu et al. [32].

2.4 Machine Learning in Packet-Level Analysis

2.4.1 Principles of Machine Learning and Deep Learning

Machine learning is a branch of artificial intelligence in which computer systems learn from data and improve their performance through experience without being specifically programmed for each individual rule. Traditional machine learning methods are typically divided into supervised learning, unsupervised learning, and reinforcement learning. This thesis focuses on supervised learning, in which a model is trained using input data and corresponding known class labels, with the aim of predicting the class for new, unknown inputs. [39] In traditional machine learning methods, such as decision trees or support vector machines, the success of classification depends heavily on feature engineering performed by experts, in which relevant features are manually extracted from raw data. [38] [39]

Deep learning is a subclass of machine learning based on artificial neural networks. Unlike traditional shallow machine learning models, deep learning models utilize multiple hidden layers to learn hierarchical representations of data directly from raw data [39]. This capacity to learn hierarchical representations, known as automated feature learning or end-to-end representation learning, eliminates the

need for manual feature design and enables the modeling of complex dependencies from large data sets [38]. Deep learning has proven effective in analyzing high-dimensional data, such as images and network traffic, where the internal structures of the high-dimensional data are complex [32].

CNNs are a deep learning architecture designed specifically to handle grid-based data, such as images. The CNN architecture typically consists of convolutional layers, pooling layers, and fully connected layers. In the convolutional layer, filters slide over the input data and perform mathematical transformations that enable the network to learn to recognize local structures and features [39]. When network traffic packets are converted into two-dimensional image matrices, CNNs can utilize spatial correlations between bytes in the same way they recognize edges or shapes in photographs [35]. However, because the 1D-CNN utilized in this thesis processes network traffic as a one-dimensional sequence, the 7×7 image matrices are employed strictly for human-readable visual inspection and interpretability analysis rather than as input for spatial correlation learning [32].

The training process of neural networks is based on minimizing the loss function. During forward propagation, the input data passes through the network, and each layer performs calculations using weight coefficients and activation functions. The difference between the prediction and the actual value is calculated using the loss function, after which a backpropagation algorithm is used to propagate the error back through the network. Optimization algorithms iteratively update the network's weight coefficients to reduce the error, allowing the network to learn to recognize meaningful patterns in the data. [35]

2.4.2 Packet Preprocessing and Sequential Representation

In this thesis, raw network traffic is converted into a numerical format that allows the use of deep learning models without traditional feature extraction. During the

preprocessing stage, network traffic capture files are decomposed into individual 802.11 frames, which are treated as independent observations regardless of their belonging to a larger traffic flow. This approach is based on the assumption that individual packets contain sufficient type-specific information for identification when processed directly as raw sequential data.

The transformation of packets into a compatible input format begins with the isolation of a specific number of bytes from the beginning of each frame. In this work, the first 49 bytes of each packet are selected to form a one-dimensional sequence. This selection is based on the fact that the header information at the beginning of the packet contains significant protocol-specific structures, such as MAC, IP, and transport layer headers, which form recurring patterns specific to devices and protocols. If the packet length is less than 49 bytes, it is padded with zero bytes to achieve the required length. Conversely, if the packet exceeds this length, the excess part is truncated so that the size of the input data remains constant. The hexadecimal value of each byte is converted to a decimal integer between 0 and 255. To make the data more suitable for neural network input and improve numerical stability, these integer values are normalized to floating-point numbers between 0 and 1.

The goal of this conversion process is to utilize the one-dimensional CNN architecture's ability to recognize local and temporal dependencies in the data without disrupting the inherent structure of the protocol stack. In network protocols, specific bytes are located at fixed positions or follow a regular linear arrangement. When these bytes are processed in their natural order as a one-dimensional vector, the relationships between protocol fields appear as sequential patterns that the convolution filters can learn to recognize. This method effectively eliminates the need for manual parsing of protocol fields and enables the direct use of raw packet data for representation learning. While the machine learning model operates on these one-

dimensional sequences, the 49-byte vectors can be reshaped into two-dimensional 7x7 matrices for human-readable visual inspection and interpretability analysis.

2.5 Related Work

2.5.1 Flow-Based Identification Methods

Flow-based detection methods analyze network traffic by aggregating individual packets into larger entities, such as TCP sessions or traffic occurring during a specific time window [8]. These methods typically rely on statistical indicators calculated from the characteristics of several consecutive packets, such as IAT, flow duration, amount of data transferred, and traffic rate. For example, the multi-stage classification system developed by Sivanathan et al. [4] utilizes flow-level attributes such as activity periods, signaling patterns, and cipher suites. The classification system achieves over 99% device identification accuracy by aggregating traffic into flows and calculating statistical features on an hourly basis, but the required time window introduces an inherent identification delay and necessitates pre-routing traffic capture at the local network edge to isolate device-specific flows. [4]

Flow-based identification methods that rely on entire TCP sessions introduce significant identification delays, as the classification model must wait for the TCP session to terminate before extracting the required feature vector [40]. To avoid the identification delay caused by full TCP sessions, Ammar et al. [40] utilize the first packets of network traffic flows combined with textual payload features to achieve near-real-time device classification. To further address the latency constraints of extensive traffic observation, Pinheiro et al. [8] propose a method based on a strict one-second time window to analyze encrypted traffic. In the one-second time window approach, statistical indicators such as the mean packet length, the standard deviation of the packet length, and the total number of transmitted bytes are calculated

to characterize the data source without inspecting the packet payload [8].

Periodic traffic analysis has also been used in device identification. Marchal et al. [7] developed the AuDI system, which models the periodic communication that takes place in the background of IoT devices without user interaction. This method does not require pre-labeled training data, but it relies on long-term monitoring of traffic flows and periodicity detection, which introduces a significant recognition delay compared to analyzing individual packets [7]. What these flow- and session-based methods have in common is that they require a stable environment, extensive packet observation, and temporal monitoring, which increases the recognition delay and requires more memory resources from network edge devices [8].

2.5.2 Single-Packet and Frame-Based Methods

Unlike flow-based approaches, single-packet or frame-based methods aim to identify devices immediately from individual observations, which is essential for real-time applications or infrequently transmitting devices. Aksoy and Gunes [33] presented the SysID system, which is capable of identifying the type of IoT device using only a single TCP/IP packet. Their method uses genetic algorithms to select the most distinctive features from the packet headers, allowing identification to be performed without the need to collect long time series or multiple packets [33]. This shows that a single packet often contains enough information to form a device fingerprint.

In IEEE 802.11 WLAN networks, the information in the wireless layer provides a significant addition to identification. Chowdhury et al. [31] have studied both packet-level and frame-level analysis in the identification of IoT devices. Their research shows that by combining network layer packet information and features obtained from IEEE 802.11 MAC frames (such as Probe Request frames), high identification accuracy can be achieved even between devices from the same manufacturer. Frame-level analysis is particularly useful in situations where a device has

not yet established a connection to an AP but is sending management frames. [31]

Recent research has moved away from manually selected features towards deep learning and direct utilization of raw data, which is in line with the approach used in this thesis. End-to-end representation learning enables automatic feature extraction directly from packet bytes without protocol-specific parsing [38]. For example, Bendiab et al. [34] developed a method in which network traffic packets are converted into RGB images and analyzed using CNNs. Their approach showed that the structural and visual features of packets can effectively reveal, for example, malicious traffic without the traditional opening of packet contents. This packet-to-image conversion leverages the ability of CNNs to learn spatial dependencies in data, providing an alternative to traditional statistical indicators.

In addition, Liu et al. [32] proposed a method in which the lengths and directions of consecutive packets form a sequence that is analyzed by a 1D-CNN. Although their method uses multiple packets, it emphasizes the importance of packet length in identifying the device and its status, especially in encrypted traffic where the payload is not readable. Consequently, analyzing individual frames and raw data via deep learning may enable faster identification and reduces reliance on manual feature engineering.

2.5.3 Limitations of Existing Methods

Although significant results have been achieved in the identification of IoT devices using machine learning, current methods have limitations that reduce their applicability in real-time and dynamic network environments. One of the most significant challenges is related to the delay of flow-based methods. Achieving high recognition accuracy often requires the analysis of entire TCP sessions or a sufficiently long observation period to calculate the statistical characteristics of the traffic flow [40] [8]. This inevitably causes a delay in the recognition process, as feature extraction can

only be performed after the session has ended, preventing an immediate response when a device connects to the network [40].

Another key limitation is the reduced effectiveness of traditional deep packet inspection and payload analysis as encrypted traffic becomes more common. IoT traffic is also increasingly encrypted, which prevents payload-based identification without decryption. Since decryption is often impossible and problematic from a privacy perspective, many current methods rely heavily on packet header information and metadata, such as packet lengths and arrival times. [8] However, relying solely on header information exposes systems to spoofing. For example MAC addresses, which have traditionally been used as device identifiers, are easy to spoof, which undermines the reliability of identification [31].

In addition, traditional shallow machine learning methods often require extensive and manual feature engineering. Experts must define in advance which traffic characteristics are relevant, making the process laborious and subjective. [39] Consequently, models relying on manually selected features struggle to automatically adapt to dynamic network changes without repeated human intervention [38].

While the majority of existing methods focus on identifying specific IoT devices or distinguishing between certain device models within a network, the preliminary step of isolating IoT traffic from general background traffic is equally important. Although some studies have addressed the binary classification of surrounding network traffic into IoT and non-IoT devices, the primary emphasis in the literature remains on fine-grained device fingerprinting [8] [4] [5]. In parallel, to minimize classification delay, research has also introduced single-packet and frame-level identification methods that avoid the latency inherent in flow-based monitoring [31] [33]. However, the combination of binary IoT traffic isolation and instantaneous single-frame analysis remains underexplored. This observation highlights the need for a general identification method to reliably separate IoT and non-IoT traffic in passive network

captures, which forms the exact methodological focus of this thesis.

3 Methodology

This chapter details the methodology used to identify IoT devices from WLAN traffic. The overall pipeline, which progresses from raw data collection to the final model evaluation, is illustrated in Figure 3.1. The methodological progression begins with the data collection process and the specification of the utilized network traffic datasets. Following data collection, the data preprocessing stage encompasses feature extraction, heavy-hitter mitigation, and device-level dataset partitioning. The preprocessed data then serves as the input for the machine learning architectures, which include the dual-branch fusion model and the two ablation models. The methodology concludes with the evaluation protocol, covering the group-aware cross-validation strategy, probability calibration, and operational threshold selection.

3.1 Data Collection Process

This chapter presents the data collection process that forms the basis of the experimental phase of the study. Network traffic collection is based on passive monitoring mode, which allows all IEEE 802.11 frames to be recorded without requiring the devices to associate with an AP. This method is necessary for the experimental setup so that the network behavior of the devices and the structural features used for identification can be observed directly from the wireless transmission path without interfering with the operation of the network.

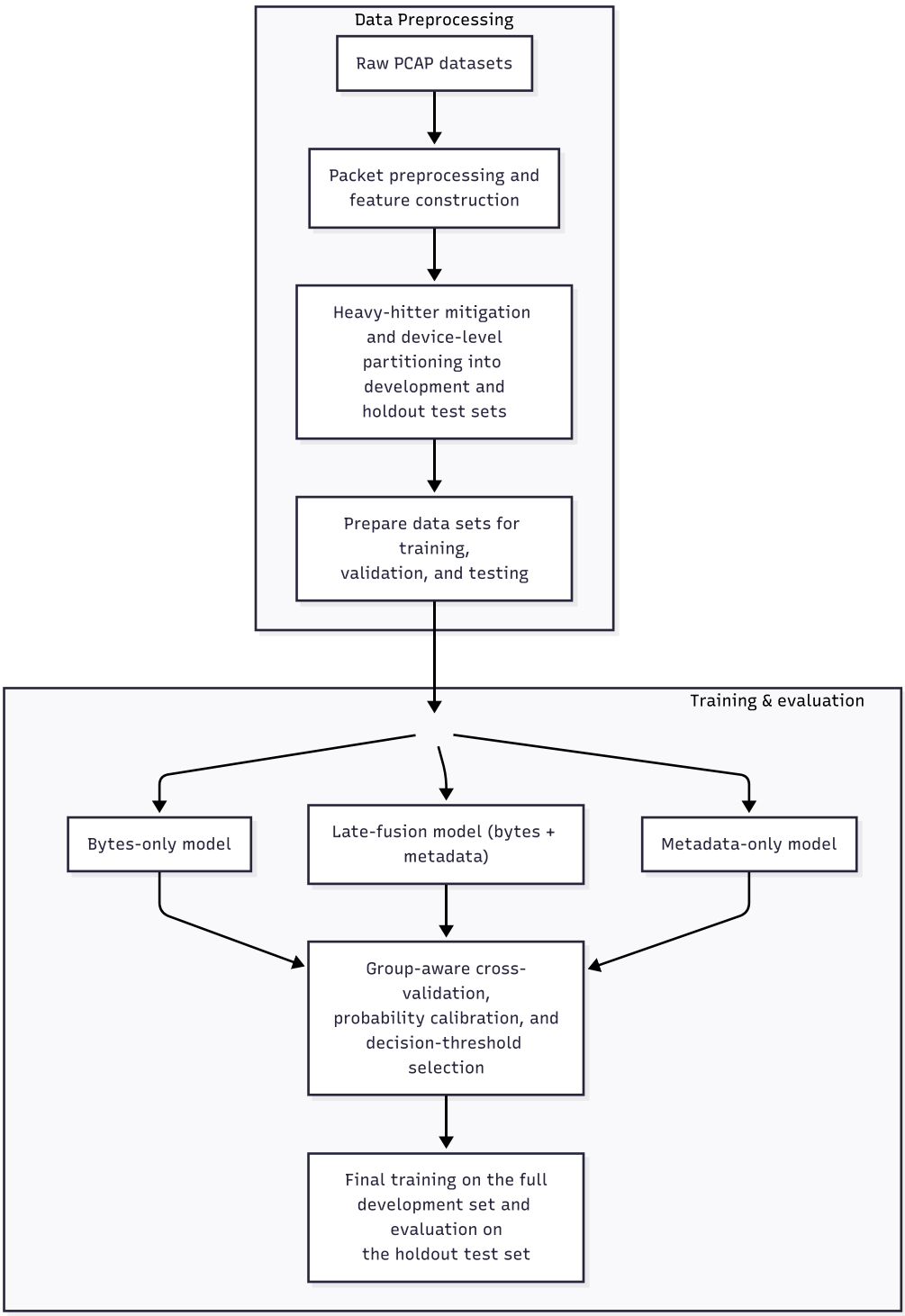


Figure 3.1: A flowchart of the methodology pipeline

The following subsections define in more detail the data sources used in the work, the structural characteristics and class distribution of the combined data set, and the privacy considerations related to data collection.

3.1.1 Data Sources and Collection Environment

The traffic analyzed in this thesis is captured passively from the wireless medium using an IEEE 802.11 WLAN network interface configured in monitor mode. This mode allows the capture of full MAC frames, including management and control frames, which are not available in standard captures performed in managed (client) mode. In managed mode, the network interface only processes traffic intended for the local device, and lower-level wireless frames such as beacon frames, probe requests, and control messages are filtered and remain invisible to the capture software. Monitor mode provides access to low-level wireless transmissions, even from devices that are not connected to the AP. This allows detecting and profiling IoT devices based on their observable wireless behavior.

The data used for model development was sourced from four IEEE 802.11 WLAN traffic datasets referred to as "IoT-environment-dataset" [41], "D-Link-IoT-dataset" [42], "etna-IoT-dataset" [43], and "probe-IoT-dataset" [44], all obtained from publicly available open data repositories. "IoT-environment" contains IEEE 802.11 frame captures from a laboratory environment including both IoT and non-IoT devices, with explicit device documentation enabling reliable labelling. "D-Link-IoT-dataset" comprises packet-level traces from multiple consumer IoT devices, each captured in isolation so that all samples represent IoT traffic. "etna-IoT-dataset" consists of IEEE 802.11 frames collected from a mixed network environment with both IoT and non-IoT devices. Labelling is based on matching MAC addresses to a set of known devices as specified in the accompanying documentation. "probe-IoT-dataset" includes IEEE 802.11 WLAN request traffic generated by smartphones and

tablets in typical user scenarios, and all samples are labelled as non-IoT.

All datasets were originally collected by the respective data providers using wireless sniffers in monitor mode to capture relevant management and data frames regardless of association status. The collection environments, as described in the respective source publications, include laboratory WLANs, controlled device-specific test setups, campus deployments, and anechoic chambers, which together provide a larger set of traffic samples for the analysis.

The key characteristics, labeling criteria, and original sizes of the four open-source datasets are summarized in Table 3.1. The original sample counts vary significantly, ranging from tens of thousands of frames in the probe dataset to nearly three million frames in the ENTA dataset. Due to the variations in original frame counts, data structures, and labeling logic, the sources require preprocessing before they can be combined for model training.

Table 3.1: Summary of the network traffic data sources used in the study.

Dataset	**Device content**	**Classification criteria**	**Original size**
IoT-Environment [41]	IoT devices and smartphones	IP address	125,182 samples
D-Link-IoT [42]	14 different D-Link IoT devices	IoT class only	560,399 samples
ENTA [43]	IoT devices and background traffic	MAC address	2,898,111 samples
Probe-dataset [44]	Smartphones and tablets	Non-IoT class only	20,343 samples

3.1.2 Dataset Structure and Class Balance

The dataset, comprised of the four separate datasets, was processed at the level of individual IEEE 802.11 frames, and each frame was assigned a binary class label to distinguish between IoT and non-IoT devices. The class labels were derived in accordance with the documentation of the original data sets, using either direct IP and MAC address mappings or the single-class nature of the data. The frames in the "IoT-environment-dataset" were classified based on IP addresses according to a predefined list, while in the "etna-IoT-dataset," identification was based on comparing MAC addresses to known device assigned MAC-addresses. The "D-Link-IoT-

dataset," which contains only IoT devices, and the "probe-dataset," which contains only smartphone and tablet traffic, were classified directly as non-IoT devices with uniform class labels without separate address comparison.

Before feature extraction and data splitting, duplicate removal was performed on the combined dataset. In this process, the raw byte vectors of the frames were compared and exactly identical frames were removed, and observations that produced conflicting class labels for the same frame were rejected. The purpose of this procedure is to prevent the machine learning model from learning completely redundant structural observations during the training phase. This also prevents identical training data from ending up in both the training data and the test data.

The original combined data consisted of a total of 3,478,853 frames. The reduction in dataset size across the preprocessing stages is summarized in Table 3.2. After removing duplicates, 1,441,967 identical frames were eliminated from the data, leaving 2,036,886 unique frames. The class distribution of this extensive data set was 35.88% non-IoT devices and 64.1% IoT devices. Since the dataset contained a significant imbalance at the device level, an upper limit on the number of packets per device was set during the stage discussed in section 3.2.3. As a result of this process, the final sample used for training and evaluating the machine learning model consisted of 83,795 frames. In this final experimental data set, non-IoT devices account for 35.06% (29,375 frames) and IoT devices 64.94% (54,420 frames), which preserves the natural proportions between the classes in the original data without artificial resampling. The final amount of usable data thus decreased significantly from the original.

Table 3.2: Evolution of data size during the preprocessing stages.

Preprocessing stage	**Total number of frames**	**IoT class**	**Non-IoT class**
Original combined data	3,478,853	71.12 %	28.88 %
After deduplication	2,036,886	64.10 %	35.88 %
After device-specific filtering	83,795	64.94 %	35.06 %

3.1.3 Privacy Concerns

Passive network traffic monitoring involves privacy challenges [8]. Limiting the analysis to the first 49 bytes of the frame and to the structural properties of the metadata reduces the amount of potentially sensitive information processed. Since IEEE 802.11 frame headers may be shorter than the selected 49-byte cut length, the input vector may also include the beginning of the actual payload. However, the method does not compromise privacy, as the study does not decrypt link layer (e.g. WPA2/WPA3) or application layer (e.g. TLS) encryption, nor does it perform semantic analysis of the payload. As a result, the payload bytes seen by the model are effectively encrypted and unreadable data. The solution ensures that the model learns to recognize the regularities of communication protocols typical of devices from header structures without revealing users' personal communications or sensitive content to the model.

The privacy of the method is further enhanced by data-level anonymization performed during the preprocessing stage, in which all MAC addresses contained in IEEE 802.11 headers are replaced with zeros. During the raw data conversion phase, the MAC address fields of each frame are masked by overwriting them with zero values ("00:00:00:00:00:00") before the frame bytes are forwarded as network input. This prevents the deep learning model from learning direct device identifiers or associating training data traffic with a specific physical device or individual. Removing MAC addresses helps the machine learning model to rely on generalizable structural features and traffic dynamics of network traffic, which not only protects user privacy but also improves the model's generalizability to new, previously unknown devices.

3.2 Preprocessing and Feature Representation

As outlined in the data preprocessing stage of the pipeline flowchart in Figure 3.1, this section describes the methods used to convert passively captured raw network traffic data into a format that can be utilized by a deep learning model. Preprocessing includes parsing capture files, extracting structural features and metadata from IEEE 802.11 frames, and correcting device-level imbalances in the training data. Since network traffic analysis is based on a multimodal model, the preprocessing stage produces two parallel representations for each frame: a normalized byte vector of the first 49 bytes of the frame and metadata describing the traffic dynamics. The purpose of these measures is to preserve the amount of essential information needed for device identification. The following subsections present the frame parsing, feature selection, and data splitting strategy based on device-specific packet count limits.

3.2.1 Parsing and Formatting of Data

During the preprocessing phase, raw packet data was converted into a numerical format suitable for machine learning models. A processing pipeline was implemented to read the network traffic files and format the individual data frames into one-dimensional byte sequences. Previous research has often reshaped packet bytes into two-dimensional matrices for image-based classification [34] [38]. However, this thesis treats the frame as a sequential data stream, aligning with the chosen 1D-CNN architecture. To standardize the input, the pipeline isolates a fixed-size sample of exactly 49 bytes from the beginning of each frame. This specific length was selected because research indicates that the most critical information for network traffic classification is located at the beginning of the frame [35]. A 49-byte sequence effectively captures the complete IEEE 802.11 MAC header, the logical link control header, and the initial fields of the network layer or management frame payload. Restricting

the input to 49 bytes avoids processing long, often encrypted application payloads that provide limited value for device identification. Furthermore, utilizing long inputs increases computational overhead and requires zero-padding for naturally short management frames, which can negatively impact model generalization.

The technical formatting of individual frames was executed as a sequential process, beginning with the removal of hardware-specific header information. Specifically, the physical layer radiotap header was stripped from the captured frames to ensure that the actual IEEE 802.11 MAC header consistently begins at the exact same position in the input sequence. This operation also ensures that the device identification method could work with different datasets, as it guarantees structural comparability of the inputs regardless of the specific capture hardware or monitoring environment. Subsequently, the first 49 bytes were extracted from each frame, and any frame shorter than this length was padded with trailing zero bytes to achieve a constant input size. The extracted byte values, which originally fluctuate within the standard 0 to 255 range, were scaled to floating-point numbers between 0 and 1 for the neural network training process. Finally, this 49-element normalized vector was expanded with a single depth dimension, forming a 49x1 input structure formatted explicitly for the 1D convolution operations. Although the model's training and inference processes operate exclusively on this one-dimensional sequential input, the vectors are subsequently converted into two-dimensional 7x7 matrices for visual analysis. This two-dimensional representation is utilized solely to enable visual inspection of the individual frames.

3.2.2 Feature Selection and Multi-Modal Representation

Instead of traditional feature extraction based on expert rules or deep packet inspection, this thesis utilizes multi-modal representation, which combines two different data sources. The method combines structural protocol information from the frame

and metadata describing the temporal dynamics of traffic into a unified feature set. A key advantage of multi-modal learning is its ability to utilize complementary information from different modalities, which improves the classification ability of the machine learning model [45]. In the context of network traffic, this means simultaneous modeling of static header information and device-specific dynamic communication rhythms.

The first feature set consists of a previously defined 49-byte sequence representing the structural protocol information of the frame. The raw bytes contained in the sequence are scaled to floating point numbers between 0 and 1, forming a numerical input for the neural network. Although this normalized byte vector can be conceptually formulated as a 7x7 square matrix for separate visual analysis, the actual machine learning model treats it as a one-dimensional structure. This approach preserves the protocol-level order of the frame as is, without the need for manual identification of separate protocol fields.

Another set of features consists of metadata describing the dynamics of network traffic, for which two commonly accepted variables were selected: IAT and total frame length. Unlike analysis based solely on the static byte structure of individual frames, the calculation of the IAT feature specifically requires consideration of the temporal order of network traffic. The arrival delay is calculated for each device by isolating the consecutive frames sent by each MAC address and measuring the time difference between them. Since the variance in arrival delays can be quite large in a network environment, the values are stabilized using a logarithmic transformation, after which they are scaled using min-max normalization according to the extreme values of the training data to a range between 0 and 1 [46]. Similarly, the total frame length is normalized by dividing it by the standard maximum specified in the IEEE 802.11 standard, which is currently 2346 bytes. Recent academic research indicates that packet length and arrival delays are among the most effective and computation-

ally efficient characteristics for distinguishing between IoT devices, which justifies their selection for this thesis [8] [31].

The combination of these two modalities is based on their complementary nature, as multimodal machine learning frameworks leverage heterogeneous data sources to improve model robustness and predictive performance [45]. The byte sequence captures the regularities of the protocols and transmission headers used by the device, while the metadata models the communication rhythm of the device and the volume of data transferred in the network environment. Consequently, combining these features into a multimodal architecture results in a stronger and more generalizable network traffic profile that is capable of distinguishing IoT devices from other network traffic even in situations where a static header structure alone does not provide sufficient discrimination.

3.2.3 Heavy-Hitter Mitigation and Data Splitting

Passively collected network traffic data typically exhibits a device-specific class imbalance, which is managed in this thesis by limiting the maximum number of frames produced by individual devices in the training data. In WLANs, certain devices, such as surveillance cameras that transmit continuous video streams or active routers, generate significantly more traffic than simple sensors that update their status occasionally [47]. Without proper control of this phenomenon, highly active devices would dominate the training data, causing the model to overfit to specific device profiles rather than learning generalizable protocol-level regularities. To prevent this, a device-specific upper limit on frame counts is applied during the preprocessing stage. This procedure ensures that no MAC address can produce more than a predefined limit of frames in the data, which balances the representativeness between devices and improves the model's ability to recognize devices in heterogeneous IoT environments.

After limiting the packet volumes, the dataset is divided into training, validation, and holdout test sets using a device-level group-aware split. The data partition relies on the unique MAC address of each device, ensuring that all network traffic generated by a specific physical device is directed to only one subset. Randomly dividing the data at the level of individual IEEE 802.11 frames would distribute traffic from the identical physical device across multiple subsets, causing data leakage between the training phase and the testing phase. Consequently, the model would learn to memorize the specific structural characteristics of the individual devices present in the training set rather than extracting the general protocol characteristics typical of the device class. Device-level isolation ensures that the model's performance reflects its true generalizability and reliability in cybersecurity applications, where the identification system must accurately recognize completely new and previously unobserved IoT devices connecting to the network.

The actual partitioning process was implemented as a randomized search algorithm that divides devices into subsets while optimizing the total number of frames and class distribution within predefined tolerances. Since the number of packets generated by different devices varies significantly despite the upper limit, simply randomly distributing MAC addresses would often lead to distorted subsets in terms of either the total number of frames or the balance between IoT and non-IoT classes. To solve this problem, the search algorithm iteratively generates several distribution options and evaluates their suitability in relation to the set target values, such as the desired size and class distribution of the test set. The algorithm calculates error scores for deviations and ultimately selects the device distribution that meets the set margins and minimizes class distortions. Although the available source datasets are challenging and there are few publicly available datasets suitable for the task, this optimization ensures that the training and evaluation sets formed are as statistically representative and comparable as possible.

3.3 Model Architectures and Classification Strategy

As introduced in the training and evaluation stage of the pipeline flowchart in Figure 3.1, this section presents the deep learning models used in this thesis and their technical architectures for network traffic device type identification. The classification strategy is based on one-dimensional CNNs and a multi-modal combination model designed to extract regularities specific to device types directly from packet byte sequences and dynamic traffic metadata. The following subsections justify the suitability of the selected model type for wireless network traffic analysis in device type identification, describe the structural design of the input data, and define in more detail the actual model architecture with its parameters and ablation models serving as benchmarks.

3.3.1 Model Type and Justification

1D-CNN was selected as the classification model for this thesis, which processes network traffic frames in their natural, sequential form. Previous research has often formulated network traffic byte sequences as two-dimensional matrices and analyzed them using 2D-CNN architectures designed for image analysis [34] [35]. However, this approach is suboptimal in terms of the structure of communication networks. The two-dimensional formulation creates artificial adjacencies and spatial dependencies between bytes in the input vector that have no logical counterpart in the original protocol-level structure. Since communication protocols, such as IEEE 802.11 MAC and transport layer headers, inherently form ordered linear byte sequences, a one-dimensional architecture directly reflects the actual structure of network traffic and processes the input in the same order as the network card receives it [32].

A key strength of a 1D-CNN in passive device type identification is its ability to

automatically learn device type-specific features directly from raw byte sequences without manual field-specific feature engineering or domain-specific rules [38]. As the convolution filters of the 1D-CNN model slide over a 49-byte input, they are able to detect local dependencies between consecutive bytes and recurring structural regularities, such as patterns formed by control flags and logical link control fields. This enables the modeling of individual communication profiles of IoT devices without the need for processing power-intensive and often inefficient deep packet inspection due to encrypted payloads [38]. Thus, the model is able to extract device-specific structural anomalies from the lowest levels of network traffic, providing a method for identifying devices in cybersecurity applications.

In addition to these structural advantages, the choice of 1D-CNN architecture is supported by its computational efficiency in passive network traffic monitoring. In cybersecurity applications, data volumes are typically large, and device identification must take place at the edge of the network in real time with as little delay as possible [8]. Compared to other deep feedforward neural networks, a convolutional architecture requires a smaller number of parameters to be trained [32]. Although deep learning models generally demand more computational resources than traditional machine learning methods, deploying the 1D-CNN model on a local network server or gateway relieves the resource-constrained IoT devices from heavy processing [32]. However, since network routers and gateways also possess constrained computing capacity, minimizing the model's parameter count through the 1D-CNN architecture remains essential for practical deployment [8]. Therefore, the 1D-CNN offers a technically scalable and practically feasible solution for real-time network traffic analysis at the network edge.

3.3.2 Dual-Branch Feature-Level Fusion Strategy and Input Representation

To model the structural regularities of the network protocols used by transmitting wireless devices, the fusion model utilizes a one-dimensional convolutional branch that takes a normalized 49-byte data sequence as input. The convolutional branch processes the linear input with consecutive one-dimensional convolutional layers, whose filters recognize local dependencies and recurring structures in the frame header fields. To stabilize the learning process and prevent overfitting, batch normalization and random dropout are applied between convolutional operations [32]. The spatial dimension of feature maps is compressed after convolution layers by combining global average pooling and global max pooling. Combining global average pooling and global max pooling consolidates the most essential structural information of the protocol headers into a coherent representation, which is then fed through a fully connected projection layer for combination.

In parallel with the structural analysis, the model processes the temporal and volume-based dynamics of network traffic using a separate multilayer perceptron network, which is fed with the previously defined two-dimensional metadata vector. This vector consists of pre-processed frame IAT and total frame length. The MLP branch consists of fully connected layers whose task is to learn the behavior profile formed by the device's communication rhythm and traffic volumes. Since the two-variable input is low-dimensional, its processing emphasizes batch normalization and regularization to prevent the network from overfitting to the numerical extremes of the training data. The feature representation produced by this branch enables the separation of devices that have a similar static header structure but differ in their network behavior and packet transmission frequency [4].

The feature representations produced by both parallel branches are finally combined into a unified vector using the dual-branch feature-level fusion principle just

before the final classification layer of the neural network. In this process, the structural model produced by the convolution branch and the dynamic model produced by the perceptron branch are concatenated into a single, broader representation that includes both modalities of the device's network traffic. The combined vector is passed through a fully connected fusion layer and regularization operations, enabling the deep neural network to learn the nonlinear interactions between these two separate sources of information [45]. The architecture's decision-making culminates in a single output cell that uses a sigmoid activation function to calculate a numerical probability for the network frame between the IoT and non-IoT classes. The two-branch fusion ensures that the final classification decision is based on the simultaneous utilization of static protocol structure and dynamic network behavior, which improves the model's fault tolerance with respect to single-modality anomalies [45].

3.3.3 Hyperparameter Optimization Strategy

The optimal structure and training parameters of the model were determined using a Bayesian optimization algorithm. This iterative search process was designed to identify the most effective configuration by evaluating different combinations of architectural and training variables. The search space included the number of convolution blocks, the base number of filters, kernel sizes, the number of nodes in dense layers, regularization strength, and the learning rate of the optimizer. The objective function of the search algorithm was configured to maximize the area under the precision-recall curve (AUPRC). AUPRC is a recommended performance metric for highly imbalanced binary classification tasks, as it evaluates discrimination capability while penalizing false positives [48].

To stabilize the training process and prevent the model from overfitting, dynamic control mechanisms were integrated into the training phase. An early stopping mech-

anism was applied to monitor the validation performance, interrupting the training if the AUPRC value of the validation set did not improve over three consecutive epochs. In parallel, the learning rate was dynamically reduced as the validation loss plateaued. This dynamic reduction supported the fine-tuning of the neural network weights during the final stages of training.

3.4 Evaluation Protocol

Continuing the training and evaluation stage of the pipeline presented in Figure 3.1, this section defines the evaluation protocol and criteria for verifying the performance of the classification models. Since passively collected network traffic data is unbalanced in terms of class distribution and the network behavior of IoT devices varies significantly, evaluating the reliability of the model requires appropriate metrics and device-level isolation between the training and testing phases. The following subsections present the stages of evaluation, the performance metrics applied, and the decision threshold calibration methods used to ensure the actual generalizability of the model in cybersecurity applications.

3.4.1 Group-Aware Cross-Validation

A reliable evaluation of the machine learning model was performed using group-aware cross-validations, in which the individual MAC addresses of physical devices were used as the basis for dividing the data into subsets. Instead of traditional packet-level random sampling, all network traffic originating from a specific device was directed in its entirety to either the training or validation set. This method ensures that the structural byte sequences of the 1D-CNN branch and the metadata variables of the perceptron branch fed into the neural network remain isolated at the device level throughout the training process. A device-aware partitioning criterion

is needed so that the evaluation metrics measure the actual generalizability of device type classification rather than the identification of individual packets contained in the training data alone.

The primary purpose of this device-level isolation is to prevent data leakage between the training and validation sets, which is a typical source of error in analyses based on pure random sampling. If frames produced by the same device, that are often strongly correlated in terms of time and structure end up in both the training and validation data, the machine learning model does not learn generalizable protocol-level regularities, but instead begins to memorize the specific traffic profiles of individual devices instead of learning [33]. This phenomenon biases the model's performance metrics upward during the validation phase, but leads to a collapse in identification accuracy in the real network environment. By grouping the data based on MAC addresses, the neural network is guided to learn more general structural features characteristic of IoT traffic, as it cannot rely on the precise device signatures it sees during the validation and training phases.

A device-aware validation strategy aims to simulate real-world conditions in wireless networks, where the monitoring system must be able to make classification decisions about devices that are connected to the network, are completely new, and are previously unknown to the model. In cybersecurity monitoring, the field of IoT devices is dynamic, and it is never possible to cover all device models on the market during the training phase. By ensuring the validation set consists exclusively of MAC addresses unseen during training, the evaluation measures the model's ability to recognize previously unseen IoT devices. The performance estimate obtained in this way reflects the operational reliability of the model when dealing with unknown devices.

3.4.2 Probability Calibration and Threshold Selection

In network monitoring and the automation of security policies, it is important that the confidence levels provided by the device identification system can be trusted, as they serve as reliability metrics for operational alarm and decision thresholds [4]. Deep learning models typically produce uncalibrated raw values, which means that their classification decisions may be either overconfident or underconfident [49]. To solve this problem, the predictions provided by the neural network were post-processed with probability calibration, which utilized predictions collected during cross-validation and isolated from the training phase. This step corrects the distortion between the values produced by the neural network and the actual device detections, so that the calibrated output reflects the actual statistical probability that the detected network frame originates from an IoT device. Since calibration is based solely on traffic isolated from the actual training data, the method aims to simulate real network environment conditions and ensures the reliability of probability estimates even when completely new devices unknown to the model connect to the wireless network.

After calculating the calibrated probabilities, the final decision threshold of the model was optimized separately, as the default classification threshold of 0.5 for machine learning models is rarely optimal for network traffic data with a highly unbalanced class distribution [50]. When there is a significant imbalance between the packet counts of the IoT device class and the non-IoT device class, using a fixed default threshold typically leads to distorted performance, with the numerically larger class dominating the classification decisions. To solve this problem, classification performance can be optimized by shifting the threshold according to the characteristics of the unbalanced data [48]. Fine-tuning the threshold allows for finding an application-specific balance between false positives and false negatives, which is critical in scenarios where prediction errors possess asymmetric cost structures, without

having to change the neural network architecture itself or its learned weights [39].

From a network security perspective, leaving a potentially vulnerable IoT device undetected poses a significantly greater security risk than allowing the detection system to produce a moderate number of false alarms [34]. For this reason, the final decision-making threshold was determined using criteria that emphasize device detection more strongly than absolute accuracy of detections. The F2 metric was applied as a methodological tool, which values the detection of all IoT devices twice as important as minimizing false alarms. In addition, a requirement was set for the system when selecting the threshold value: the detector must guarantee a detection rate of at least 80% for all IoT traffic. Subject to this threshold being met, the threshold value was fine-tuned to a level that minimizes unnecessary false positives. This strategy ensures that users of the model receive a sufficiently comprehensive and reliable situational picture of the devices connected to the network, while at the same time keeping the model usable without an overwhelming flood of alerts [51].

4 Implementation and results

4.1 Development environment setup

The training of the machine learning models and the data processing in this thesis were conducted in a local development environment. Detailed hardware and software specifications are summarized in Table 4.1.

Table 4.1: Development environment and software stack

Category	Component	Specification
Platform	Processor	AMD Ryzen 7 7700
	Graphics Processor	AMD Radeon RX 9070 XT 16 GB
	Memory	32 GB DDR5
	Operating System	Ubuntu 24.04 LTS
Software Stack	Programming Language	Python 3.12
	Core Libraries	TensorFlow, Keras, Scikit-learn, NumPy, Pandas

A dedicated GPU was utilized to accelerate the training of the models on network traffic datasets. Data preprocessing, feature extraction, and model evaluation were implemented in Python using standard data science libraries. To ensure the reproducibility of the research, the random seeds for all software libraries used were fixed globally to constant values prior to the model training phase.

4.2 Implementation of the Classification Models

This section presents three classification models to be compared in an experimental setting: structural and dynamic traffic data. A dual-branch feature-level fusion model, which serves as the main model, and two ablation models derived from its architecture (bytes-only and meta-only). The training phase for all models is performed on a training set using a consistent, device-aware cross-validation protocol. In this implementation, the raw data is processed as a one-dimensional 49-byte sequence, which reflects the sequential structure of frame protocols and is suitable for 1D-CNN architecture. The following subsections describe in more detail the structure of each model, the inputs they use, and their role in evaluating the discriminative power of individual features.

4.2.1 Dual-Branch Feature-Level Fusion Architecture

Based on the model architecture presented in Chapter 3, the final structure and parameters of the fusion model were determined using hyperparameter search. As a result of this process, the model's capacity, such as the number of convolution blocks and layer sizes, was adjusted to meet the requirements of the network traffic characteristics. The detailed architectural structure of the fusion model is presented in Table 4.2, and the final hyperparameters of the hyperparameter search are summarized in Table 4.3.

The training phase of the model was performed on a training set using device-aware cross-validation. The detailed training and validation configuration is summarized in Appendix A.1. Device-aware partitioning ensures that traffic from a single MAC address does not occur simultaneously in the training and validation data, which guides the model to learn protocol-level regularities specific to IoT devices rather than simply memorizing device-specific anomalies.

Branch	Stage	Layers (in order)	Output shape
Bytes	Input	Input: 49×1 sequence	$(None, 49, 1)$
Bytes	Conv block 1	Conv1D(64) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPool $\rightarrow$ Dropout	$(None, 24, 64)$
Bytes	Conv block 2	Conv1D(128) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPool $\rightarrow$ Dropout	$(None, 12, 128)$
Bytes	Conv block 3	Conv1D(256) $\rightarrow$ BatchNorm $\rightarrow$ ReLU $\rightarrow$ MaxPool $\rightarrow$ Dropout	$(None, 6, 256)$
Bytes	Pooling	GlobalAveragePool1D $\|$ GlobalMaxPool1D (concatenate)	$(None, 512)$
Bytes	Projection	Dense(64)	$(None, 64)$
Metadata	Input	Input: 2 scalar features	$(None, 2)$
Metadata	MLP block	Dense(16) $\rightarrow$ BatchNorm $\rightarrow$ Dropout	$(None, 16)$
Fusion	Concatenation	Concat(Bytes proj, Meta embedding)	$(None, 80)$
Fusion	Classifier head	Dense(96) $\rightarrow$ Dropout $\rightarrow$ Dense(1, sigmoid)	$(None, 1)$

Table 4.2: Model architecture. The network consists of a 1D-CNN branch for the byte sequence and a small MLP branch for metadata, fused by concatenation and followed by a dense classification head.

Hyperparameter	Search space	Best
`cnn_blocks`	Integer {1, 2, 3} (default 2)	3
`filters_base`	Integer {32, 64} (step 32)	64
`kernel_size`	Choice {3, 5, 7}	3
`cnn_dropout`	Float 0.1–0.4 (step 0.1)	0.3
`bytes_projection`	Integer {16, 32, 48, 64} (step 16)	64
`meta_units`	Integer {8, 16, 24, 32} (step 8)	16
`meta_dropout`	Float 0.0–0.2 (step 0.1)	0.1
`fusion_units`	Integer {32, 64, 96, 128} (step 32)	96
`dropout_fusion`	Float 0.2–0.5 (step 0.1)	0.3
`learning_rate`	Choice {1e-3, 5e-4, 1e-4}	5e-4

Table 4.3: Hyperparameter optimization setup and best configuration (Bayesian optimization; objective: validation AUPRC).

The cross-validation results showed that the model achieved optimal accuracy very quickly, typically after only two or three passes. This rapid learning is consistent with the nature of network traffic: the header structures and traffic dynamics of 802.11 frames generated by IoT devices are largely repetitive and standardized compared to the irregular network traffic of traditional user devices. The limited availability of source data also contributes to rapid learning.

Based on the results collected from cross-validation, the final number of training epochs for the fusion model was set to three. In accordance with this setting, the model was retrained once using the entire available training set. The use of all training data ensures that the classifier has access to the maximum number of samples during the learning phase in order to form as broad and representative an IoT traffic profile as possible before the model is transferred to a separate test set for evaluation.

4.2.2 Bytes-Only Ablation Model

The first test benchmark is an ablation model that processes only raw frame data, with the aim of measuring the independent classification ability of 802.11 header structures alone, without metadata representing network traffic dynamics. The model utilizes the one-dimensional convolution branch of the previously presented combination model, which in this setup is isolated as an independent classifier. The input consists solely of a normalized sequence of 49 elements consisting of the first bytes of the frame. The sequence is processed by convolution and normalization layers, whose feature maps are condensed and fed directly to the classification layer without parallel feature sources. The goal of the architecture is to test how reliably IoT devices can be identified using only the byte data of the devices and ultimately compare it to the fusion model. The structure of this ablation model is summarized in Table 4.4.

Stage	Layers (in order)	Output shape
Input	Input sequence 49×1	`(None, 49, 1)`
Conv block 1	Conv1D(64) → BatchNorm → ReLU → MaxPool → Dropout	`(None, 24, 64)`
Conv block 2	Conv1D(128) → BatchNorm → ReLU → MaxPool → Dropout	`(None, 12, 128)`
Conv block 3	Conv1D(256) → BatchNorm → ReLU → MaxPool → Dropout	`(None, 6, 256)`
Pooling	GlobalAveragePool1D ‖ GlobalMaxPool1D (concatenate)	`(None, 512)`
Projection	Dense(64, ReLU)	`(None, 64)`
Classifier head	Dense(96, ReLU) → Dropout → Dense(1, sigmoid)	`(None, 1)`

Table 4.4: Bytes-only ablation model architecture. The network consists of a 1D-CNN branch for the byte sequence followed by global pooling, a dense projection layer, and a dense classification head.

To ensure consistent comparability, the training of this trimmed model was performed using exactly the same device-aware cross-validation protocol and training set as in the fusion model acting as the main classifier. The fine-tuning parameters of the training phase, such as the optimization algorithm, loss function, and early stopping conditions, were kept completely identical to those presented in Appendix A table A.1. Maintaining a constant evaluation protocol and identical training data ensures that any performance differences observed during the testing phase result exclusively from the constrained input representation. Isolating the input representation enables a direct evaluation of the discriminative power of protocol-level raw bytes compared to the temporal dynamics of the network traffic.

4.2.3 Meta-Only Ablation Model

The second experimental benchmark is an ablation model that deals exclusively with network traffic metadata, with the aim of evaluating the discriminatory power of computational metadata without analyzing the structural content of the frame. The input for this classifier is a two-dimensional feature vector consisting of packet IAT

and frame length. The model is separated from the main model architecture, utilizing only its multilayer perceptron network, which processes low-dimensional input through densely connected layers, batch-wise normalization, and dropout layers that suppress overfitting, all the way to the final classification node. The purpose of the architecture is to force the model to make classification decisions based purely on the network behavior of the device, i.e., the transmission rhythm and data volume of messages. The structure of this metadata-based model is shown in Table 4.5.

Stage	**Layers (in order)**	**Output shape**
Input	Input: 2 metadata features	`(None, 2)`
Metadata embedding	Dense(16, ReLU) → BatchNorm → Dropout	`(None, 16)`
Classifier head	Dense(96, ReLU) → Dropout → Dense(1, sigmoid)	`(None, 1)`

Table 4.5: Metadata-only ablation model architecture. The model consists of a small MLP operating on two metadata features followed by a dense classification head.

As in the previous ablation model, the training and performance evaluation of this metadata-based classifier was performed in accordance with a consistent device-aware cross-validation protocol with the main model. The training phase applied the same training sets, hyperparameter settings, and other conditions previously defined in Table A.1. By keeping all other variables constant except for the input data, the test setup ensures comparability. This restriction ensures that any differences between the models observed later in the evaluation phase can be directly attributed to the lack of structural information at the protocol level, which allows for the measurement of the identification value of network traffic dynamics alone.

4.3 Evaluation and Results

This section summarizes the numerical and qualitative results of the testing phase of the classification models trained in the setup. The models are evaluated in their

entirety using a holdout test set so that the results reflect as reliably as possible the actual generalization ability of the models in network traffic generated by previously unknown devices. The first subsection compares the classification performance of a fusion model combining structural and dynamic traffic data with pruned ablation models in terms of key classification metrics. This is followed by an analysis of the reliability and robustness of the models by evaluating their ability to tolerate feature shifts between the training and test sets and by examining the calibration of prediction probabilities. Finally, the numerical results are deepened with visualizations that allow the model's decision-making logic and misclassifications to be interpreted directly at the level of individual 802.11 frame features.

4.3.1 Classification Performance on Holdout Data

The classification ability of the fusion model and the ablation models was evaluated on an isolated test set using threshold-independent metrics. This approach objectively compares how reliably different feature sources can distinguish IoT traffic from other network traffic before a specific decision threshold is set. The models' capability to balance true positives and false positives was evaluated using the ROC curve, whereas the PR curve was utilized to examine classification performance in the context of the unbalanced class distribution. In the PR evaluation, the baseline for random guessing is established at 63.1%, corresponding to the actual proportion of IoT frames in the test data. The inherent discrimination power of the models is visualized in figure 4.1. The comparative results show that the bytes-only model, which analyzes only raw protocol-level data, achieves AUROC value of 96.4%. This performance is on par with the fusion model that utilizes both feature sources, which achieves an AUROC of 96.3%. In contrast, the meta-only model, which relies solely on network traffic dynamics, performs significantly worse with an AUROC of 84.0%. A similar phenomenon is observed in the AUPRC metric, where the models utiliz-

ing raw data reach nearly 98% (the fusion model achieves 98.1% and the bytes-only model achieves 97.9%), while the meta-only model remains at 89.5%. These findings confirm that the structural content of IEEE 802.11 frame headers provides a substantially stronger signal for device class separation than dynamic metadata based solely on message rhythm and size.

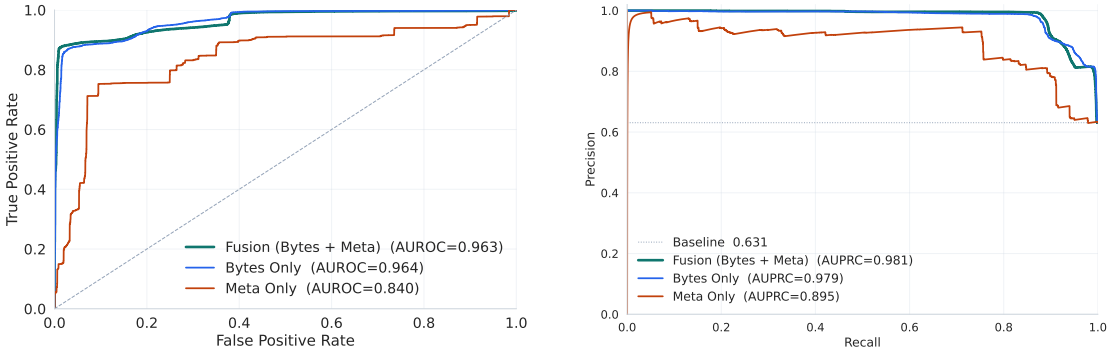


Figure 4.1: Comparison of the native discrimination power of the fusion model and the ablation models using the holdout test set. The ROC curve (left) and the PR curve (right) illustrate the threshold-independent classification performance of the different feature representations.

To verify the operational recognition capability required by security applications, accurate classification metrics were calculated for the models using the decision threshold defined in the training phase, which emphasizes a high detection rate. Although the ablation models showed good inherent discrimination power, their operational performance at a fixed threshold reveals the limitations of individual feature sources. The fusion model achieves the highest overall performance on the isolated test set with an F1 score of 90.5%, while the bytes-only model 79.7% and meta-only model 78.9% fall significantly short of the target level. The most critical difference in terms of network security is observed in the recall metric, which measures the detection rate of IoT frames. The bytes-only model, which analyzes solely raw frame data, correctly identifies only 66.7% of the IoT traffic. In a practical monitoring system, this deficiency would result in a significant volume of unidentified IoT frames. The metadata-based model, on the other hand, achieves a recall

rate of up to 94.0%, but its extremely low precision of 68.0% would generate too many false alarms from normal user devices in a real-world environment. Only the fusion model, which combines source data, is able to maintain a high accuracy of 99.5% while achieving a detection rate of 83.1%. The accurate classification metrics are compiled in a comparison table 4.6 and presented in figure 4.2.

Table 4.6: Model performance on the holdout test set

Model	AUROC	AUPRC	Precision	Recall	F1-score	Accuracy
Fusion-model	96.34%	**98.07%**	**99.47%**	83.07%	**90.52%**	**89.03%**
Bytes-only	**96.38%**	97.94%	99.13%	66.66%	79.72%	78.60%
Meta-only	83.96%	89.53%	67.97%	**93.99%**	78.89%	68.27%

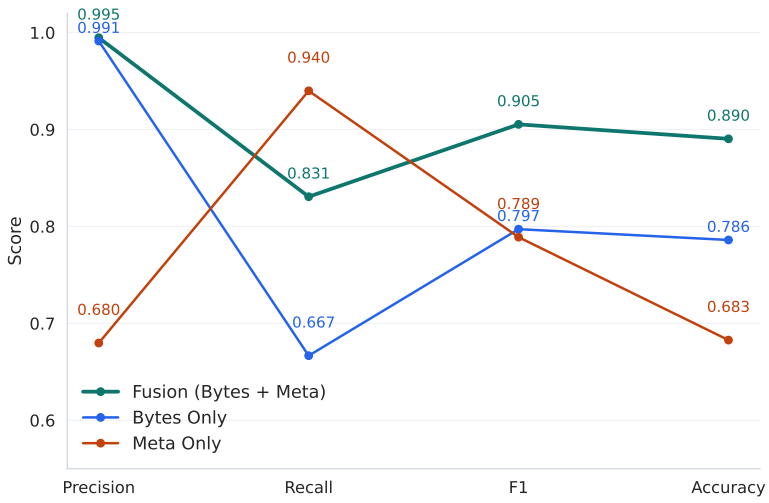


Figure 4.2: Operational classification performance profiles of the fusion model and ablation models at a decision threshold set during the training phase on a test set. The graph illustrates the differences between different data sources in terms of the model's ability to avoid false alarms (precision), detect IoT devices in the network (recall), balance the relationship between the two (F1), and correctly classify all traffic under inspection (accuracy).

A closer look at the misclassifications made by the fusion model shows that the fusion of static structural information and dynamic traffic behavior improves the conditions for reliable device identification in a real-world heterogeneous network environment. The confusion matrix of the classification shows that the hybrid model

is able to accurately identify non-IoT traffic in the test set, producing an incorrect IoT classification for only 0.8% of normal frames. This low false alarm rate is an important feature for minimizing load, for example, in cybersecurity management systems. On the other hand, the model fails to recognize 16.9% of IoT device frames. Such a margin of error is to be expected, especially with such limited source data, because in wireless traffic, devices occasionally send standard 802.11 control and management frames that are structurally and temporally very similar to the traffic generated by normal user devices. Despite the proportion of unidentified frames, the model's ability to distinguish traffic with high accuracy shows that the porposed model provides sufficient observation accuracy for security monitoring purposes. The confusion matrix of the combined model with the test set is shown in Figure 4.3.

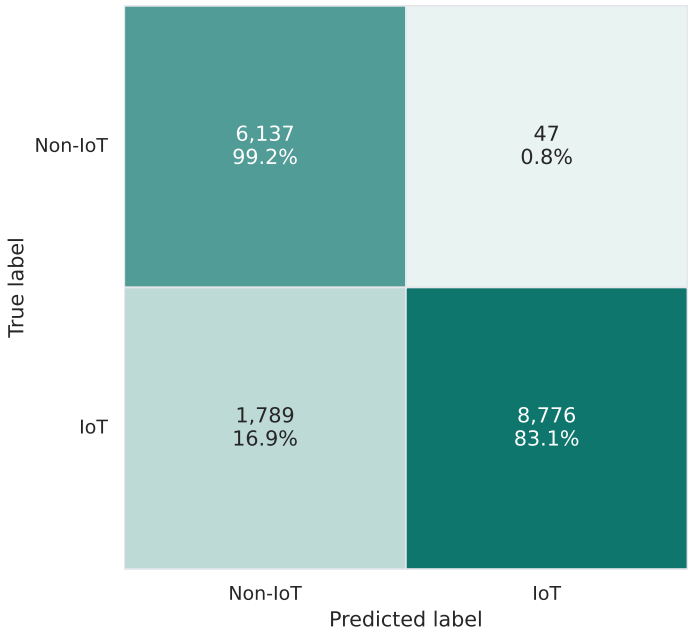


Figure 4.3: Confusion matrix of the fusion model using the holdout test set. The cells represent the absolute number and the percentage of classified IEEE 802.11 frames relative to their actual device classes.

4.3.2 Model Robustness: Calibration and Feature Shift

Due to the dynamic nature of network traffic and the limited source data available for this thesis work, there is a feature shift between the training and test sets, which poses challenges for the generalization ability of machine learning models in recognizing new devices. The data used in the training phase contained a clear statistical difference between device classes, as the average IATs and frame lengths of IoT devices and other traffic differed significantly. However, when moving on to evaluate the model with the test set, these distinguishing features weakened significantly, and the average differences between the classes narrowed to only a small fraction of those observed in the training data. According to this strong overlap in feature distributions, in the unseen test data, the traffic sent by IoT devices is much more difficult to distinguish from the network traffic produced by normal user devices in terms of its temporal rhythm and data volume. This similarity in traffic dynamics makes it difficult to distinguish between devices based on statistical features alone. The shift in distributions and overlap between classes are illustrated in the feature-specific figures 4.4 and 4.5.

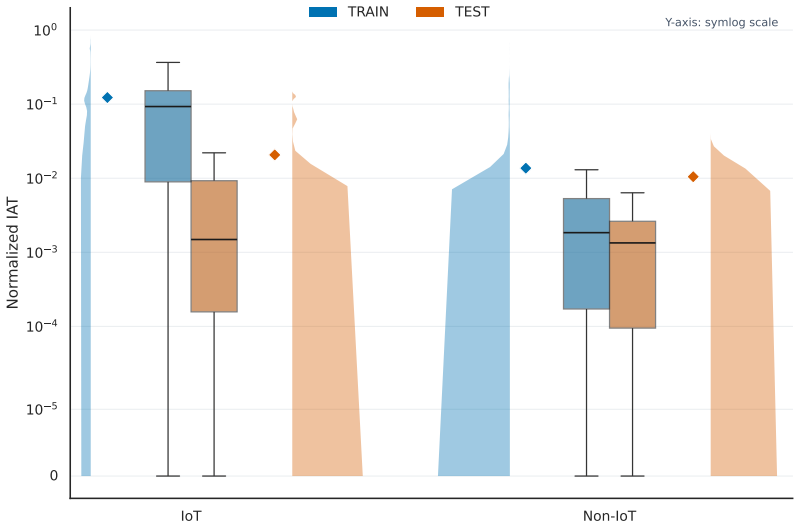


Figure 4.4: Normalized distributions of IATs in the training and test sets, presented on a logarithmic scale.

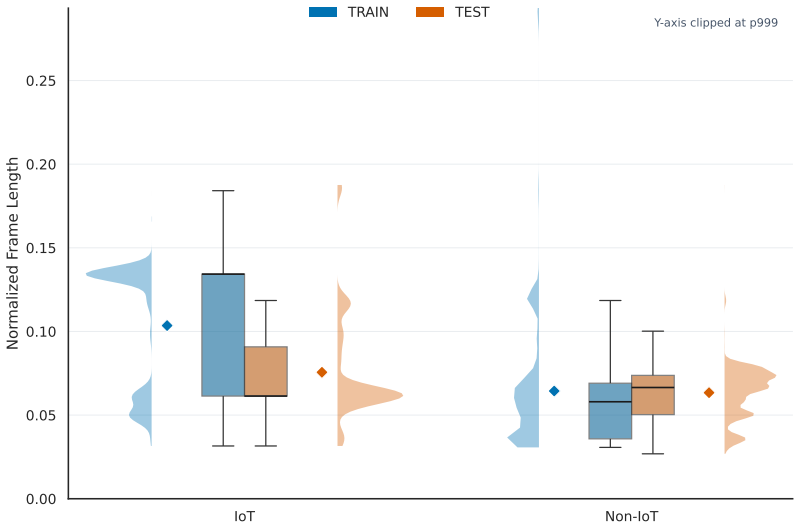


Figure 4.5: Distributions of normalized frame lengths in the training and test sets.

The observed differences in traffic profiles are reflected in the robustness of the classifiers, and the analysis shows the weakness of a model based solely on metadata in dynamic conditions. Although the overall performance of all models, such as calibrated AUROC, independent of the threshold value, even improved relatively when moving from the training set to the test set, the shift in feature distributions led to significant problems in classification for the metadata-based model. Since the IATs and lengths of IoT and non-IoT traffic overlapped significantly in the test data, the meta-only model produced a large number of false alarms at the decision threshold set in the training phase, which dropped its classification precision to a low 68.0%. In contrast, the bytes-only and fusion model, which utilize raw data, were able to tolerate the shift in metadata much better, as their decision-making is based on the protocol structure of the frame. This confirms that traffic-based detection requires structural analysis to be robust in variable network environments. The detailed performance change between cross-validation and the test set is presented in the comparison table 4.7.

In addition to actual classification ability, it is important in cybersecurity monitoring systems to assess the reliability of the prediction probabilities provided by

Table 4.7: Relative change in calibrated discriminative performance (AUROC and AUPRC) when moving from out-of-fold validation to the holdout test set.

Model	OOF AUROC	TEST AUROC	AUROC Δ	OOF AUPRC	TEST AUPRC	AUPRC Δ
Late-fusion	91.70%	96.30%	5.10%	93.40%	98.10%	5.00%
Bytes-only	91.20%	96.40%	5.70%	92.60%	97.90%	5.80%
Meta-only	71.10%	84.00%	18.00%	78.70%	89.50%	13.70%

the model, which was measured in the comparison setup using the expected calibration error (ECE). In an optimal situation, the probability provided by the model directly corresponds to its empirical accuracy, meaning that, for example, 90% of IoT classifications made with a probability of 90% are, on average, correct in reality. The results show that the fusion model of the two feature sources is clearly the most reliable in its estimates, as its calibration error on the test set is only 19.4%. The calibration error of the bytes-only model, which uses only raw data, is 23.3%, while the model based solely on metadata is significantly less reliable, with an error rate of 30.7%. The lower calibration error of the fusion model means that the confidence levels it produces can be relied on more reliably in automated information security management mechanisms, where the risk of false alarms must be minimized. The relationship between the models' predicted probabilities and empirical accuracy is shown in the calibration figure 4.6.

4.3.3 Model Interpretability and Decision Logic

To verify the reliability of the classification results and the quality of the information learned by the model, the feature space formed by the fusion model was examined using a two-dimensional UMAP projection illustrating the grouping of device classes. The two-dimensional projection shows that during its learning process, the model is able to distinguish between IoT device traffic and regular traffic as clearly distinguishable clusters even before the final classification decision. This confirms that the neural network has learned regularities in traffic rather than simply memorizing

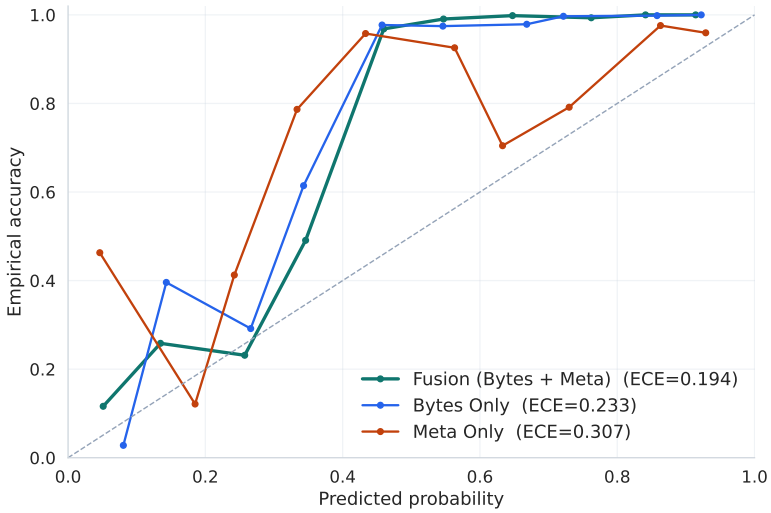


Figure 4.6: Reliability diagrams for the fusion model and the ablation models using the holdout test set. The diagram illustrates the relationship between the predicted probabilities and the actual empirical accuracy, where the diagonal dotted line represents optimal calibration.

training data. When the misclassifications made by the model on the test set are placed in the same projection, they are found to be concentrated almost exclusively at the boundaries between clusters or in separate islands where the characteristics of different device classes overlap. This overlap reflects the true nature of network traffic, where standard frames sent by certain IoT devices can be almost identical in structure to traffic generated by regular user devices. The classification of the drawing space and the location of errors are illustrated in Figure 4.7.

In the context of security, it is useful to understand the decision-making logic of the model in addition to its numerical performance. This was illustrated using byte-level saliency maps of 802.11 frames. Saliency maps are calculated by determining the gradient of the classification result in relation to each element of the 49-byte sequence given as input, revealing how strongly each individual byte has influenced the decision made by the neural network. The visualizations show that the model does not always evaluate the input evenly, but rather focuses strongly on individual, precisely defined MAC header bytes. This suggests that the model may emphasize

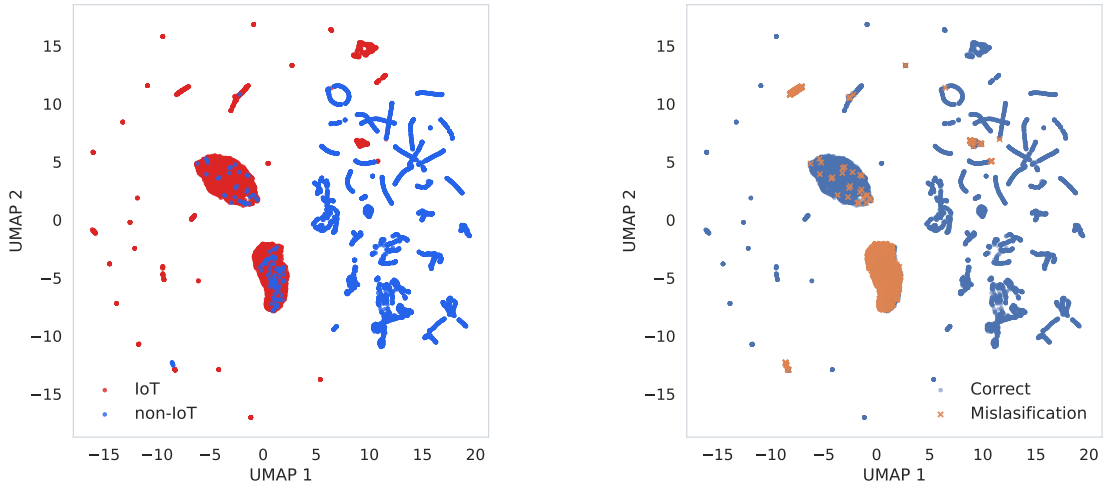


Figure 4.7: Two-dimensional UMAP projection of the latent space formed by the fusion model using the holdout test set. The left-hand graph visualizes the actual device classes of the IEEE 802.11 frames, while the right-hand graph highlights the misclassifications made by the model in relation to the correct predictions.

only a few or even a single byte when making its classification decision. Since it is practically impossible to manually review all thousands of frames in the test set, the model's overall decision-making logic was evaluated using random samples of several frames. The samples revealed that in correctly classified cases, the model repeatedly finds clear structural fixed points. This classification logic is illustrated by a sample of six frames in Figure 4.8.

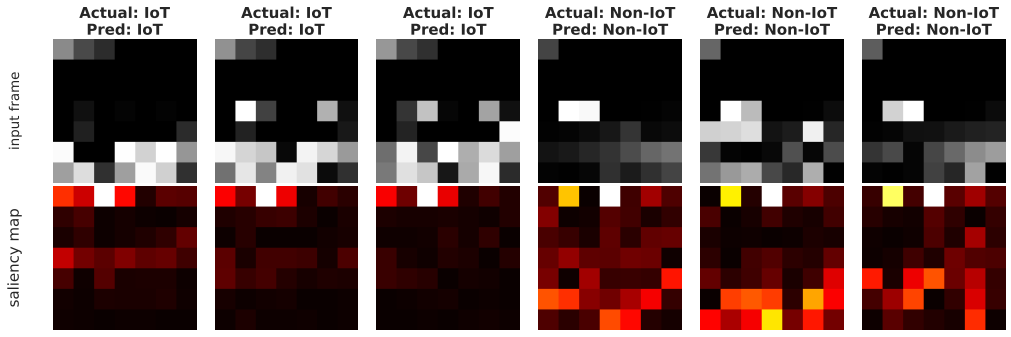
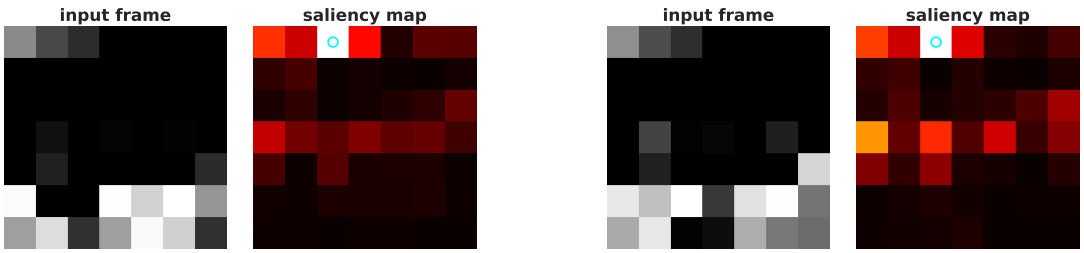


Figure 4.8: Saliency maps of six correctly classified IEEE 802.11 frames. The top row displays the first 49 bytes of each frame as grayscale images, where the first three represent IoT traffic and the subsequent three represent non-IoT traffic. The bottom row presents the corresponding heat maps, where pixel brightness indicates the impact of the respective byte on the fusion model's classification decision.

Similarly, examining incorrect classifications reveals the protocol structures that cause uncertainty for the fusion model. By comparing the attention maps of successful and unsuccessful classifications, recurring patterns were observed where the fusion model's attention focused heavily on the identical MAC header fields regardless of the final classification result. For example, the fusion model frequently reacted strongly to the third byte of the MAC header, which defines the QoS Data subtype within the IEEE 802.11 Frame Control field. Because both IoT devices and traditional user devices widely use QoS Data frames for data transmission, the structural signal provided by the QoS Data subtype byte is difficult to interpret for device class separation. Although limited random samples do not generalize the behavior of the entire holdout test set, the saliency map observations indicate that the classification errors are related to the standardized nature of IEEE 802.11 protocols and the overlapping structural characteristics of the device classes. An example of the structural ambiguity between a correct classification and an incorrect classification is presented comparatively with a saliency map in Figure 4.9. The classification challenges are also partially attributable to the limited amount of training data.



(a) Correct classification (b) Misclassification

Figure 4.9: Attention map visualization of randomly selected correctly and incorrectly classified IEEE 802.11 frames. The heat maps presented alongside the grayscale images illustrate the impact of individual bytes on the model's classification decision, with the blue circles highlighting the third byte of the frames corresponding to the QoS Data subtype.

5 Discussion

This chapter evaluates the practical applicability of the proposed device identification method within the context of wireless network security. The discussion assesses how the dual-branch fusion architecture meets the requirements of cybersecurity monitoring, with a focus on the structural ambiguity of the IEEE 802.11 protocol that naturally limits single-frame classification accuracy. Furthermore, the methodological limitations of the study are addressed, examining the constraints of the utilized network traffic datasets and the challenges introduced by feature shifts in wireless network environments. Finally, the chapter outlines recommendations for future research, including the need for more diverse data sources and potential enhancements to the classification architecture.

5.1 Implications for Network Security and Monitoring

The performance metrics presented in Chapter 4 show that the developed machine learning model is capable of distinguishing frames sent by IoT devices from other network traffic. However, statistical and numerical discrimination alone is not sufficient to verify the viability of the method in real-world cybersecurity and network monitoring applications. In order for passive observation-based detection to be reliably utilized as part of network management, the results achieved must be examined

in terms of their applicability and the limitations set by the IEEE 802.11 network protocol itself. The following subsections evaluate the practical functionality of the recognition model and analyze how the standardized structural ambiguity of network traffic limits the reliability of classification based on individual frames in dynamic network environments.

5.1.1 Operational reliability and false alarm management

The practical functionality of security systems, such as intrusion detection systems, requires the minimization of false alarms [38]. The low false positive rate of 0.8% achieved in the study is a good result in terms of this objective. In a typical network environment, conventional client devices generate significantly more traffic than IoT devices. If the detection model made errors in even a small portion of this large data mass, the system would quickly generate too many false alarms. Continuous false alarms burden maintenance, which may result in actual threats going unnoticed [51]. The threshold value of the developed model was therefore optimized primarily to protect normal traffic. This approach shows that the method works as a technical concept and is suitable for the real network environments it was designed for.

In addition to the classification decision itself, the probabilities generated by the model must also be reliable. This is a prerequisite for the recognition to be used automatically in network management without constant human supervision. In the study, the expected calibration error of the model was 0.194, which means that the confidence reported by the model corresponds with sufficient accuracy to its actual prediction accuracy. If the system identifies a device as belonging to the IoT category with a very high degree of certainty, it can automatically restrict the device's data traffic or move it to an isolated part of the network. Such automatic protective measures reduce the need for manual work by the network administrator. [7] Although the model has been implemented with limited resources as an experimental

concept, the calibration level achieved demonstrates the potential of deep learning as part of autonomous access control.

5.1.2 Structural ambiguity in network traffic

The overlap between device classes that emerges in the UMAP projection formed by the model is a direct consequence of the standards set by the IEEE 802.11 protocol. The standards force the frames of different devices to be structurally consistent. For example, the network management frames sent by an IoT sensor and a smartphone can be almost identical at the bit level. It is impossible to distinguish between such frames based on structure alone. The structural ambiguity dictated by the IEEE 802.11 protocol is clearly visible in the classification results. The two-dimensional UMAP projection in Figure 4.7 demonstrates that IoT and non-IoT device classes merge into shared clusters when the neural network fails to extract distinguishing features from the frames. Furthermore, the saliency map analysis in Figure 4.9 confirms that the classification model frequently relies on identical standardized header fields to make decisions, regardless of the transmitting device type.

The natural limitation of single-frame analysis-based detection is most evident in shared MAC layer structures such as QoS Data frames. The structural ambiguity of QoS Data frames is a basic feature of the IEEE 802.11 standard itself, not a shortcoming of the machine learning model. As Figure 4.9 illustrates, the model makes incorrect classifications when relying on bytes that define the frame subtype. In modern networks, many devices designed for completely different purposes use the same QoS header fields to prioritize traffic. Because QoS features require prioritization information to be expressed in a specific way, the header cannot contain information that reveals the identity of the device. It is due to these shared structural characteristics that perfect identification accuracy at the individual frame level is an unachievable goal in wireless networks. The developed experimental concept

has clear discrimination capabilities, but protocol standardization sets an upper limit on accuracy.

5.2 Limitations of the study

The evaluation of the developed machine learning model and the generalizability of the results are significantly limited by the lack of openly available, extensive, and diverse traffic data sets. Public data sets often contain only a limited sample of devices, or they have been collected under controlled laboratory conditions. As a result, the training data used does not perfectly model the constantly changing range of devices in the real world. A limited selection of devices may lead to the model learning to recognize only the specific features of certain manufacturers, rather than generalizing the rules to all IoT devices in the network.

Test results show that the model's performance is susceptible to feature shift, which reduces recognition reliability when moving from a static training environment to changing real-world networks. As revealed in the analysis in Chapter 4, features that were clearly distinguishable during the training phase, such as the IATs and lengths of individual frames, became significantly mixed in the test set. This change in data characteristics means that a model based solely on the traffic dynamics of a single frame loses its recognition accuracy in new network conditions. Although methods based on statistical analysis of multiple packets or time windows have been found in the literature to achieve high recognition accuracy, at the level of a single frame, statistical dynamics alone are not sufficient to guarantee good performance [8] [36] [4]. In this thesis, structural analysis of frame header fields utilizing a fusion model was a necessary requirement to compensate for the weakening of statistical features.

From a methodological point of view, selecting a single frame as the unit of analysis allows for simple analysis without examining individual protocol fields, but

makes classification more susceptible to protocol-level variations compared to analyzing long-term traffic flows. The analysis of a single frame was chosen so that devices could be identified without the need to track entire TCP/IP flows. This makes the model sensitive to momentary abnormalities in device operation and to the limited information contained in a single frame. If identification were based on the analysis of several consecutive frames or the entire session, occasional misclassifications could be filtered out more effectively. However, this would result in the loss of the ability to identify devices whose network traffic is very infrequent.

The physical topology of the network acts as an absolute constraint on identification based on IEEE 802.11 WLAN frame analysis, as the method is unable to detect sensors operating behind gateways. Many low-power IoT devices communicate wirelessly via Zigbee or BLE protocols to a separate gateway, which then connects to the actual IP network. Passive IEEE 802.11 WLAN traffic monitoring in such an architecture only detects the gateway frames and the traffic they generate, not the original transmissions from individual sensors. This is a limitation imposed by the network technology itself. It cannot be overcome through the development of a machine learning model alone, as the original signal from the sensors never reaches the IEEE 802.11 WLAN interface.

5.3 Recommendations for future work

In future research, similar classification models should be trained and evaluated using more diverse source data. The current results are based on limited source data from laboratory environments, which do not fully reflect the traffic of real-world WLANs. Labeled datasets should include a wider variety of both IoT and non-IoT devices, enabling machine learning models like those in this thesis to be trained more reliably and, consequently, better evaluated for their suitability in real-world applications.

To address the bottleneck of manually labeling large volumes of real-world network traffic, future monitoring systems could incorporate semi-supervised learning techniques. Semi-supervised learning algorithms utilize the vast amounts of unlabelled traffic data alongside a small set of initially labeled traffic data, thereby reducing the manual labeling effort. Furthermore, the continuous introduction of new IoT device models requires classification architectures that adapt dynamically. Active learning, a specific machine learning method, provides a mechanism where the monitoring system automatically isolates frames with low classification confidence and selectively requests labels from network administrators. Implementing active learning would facilitate the maintenance of up-to-date device class profiles in dynamic networks without requiring exhaustive manual labeling.

To overcome the limitations caused by the structural similarity of individual frames, one possible direction for development is to enrich the metadata utilized by the model. The standardized structure of the headers in the beginning of the frame, as well as the sensitivity of packet length and IAT to environmental factors, cause network traffic from different devices to appear similar. The use of more than two computational features can be expected to improve the discrimination power of the two-branch model developed in this work, while still keeping the analysis relatively lightweight compared to monitoring that maintains the state of entire network connections.

Since the method developed in this thesis primarily serves as an academic proof of concept of the structural discriminative power of IEEE 802.11 frames, a topic for further research could be the evaluation of the scalability and computational requirements of this approach. Although the two-branch neural network architecture achieves high recognition accuracy in the test environment, extending the concept as part of broader network monitoring would require more precise information on the load imposed by the method in practical systems. Future work should investigate

how the computational load of parallel neural networks increases as the volume of traffic to be analyzed grows. In addition, further research should examine how the classification data generated by the model could be best utilized in the field of cybersecurity.

6 Conclusion

The increase of IoT devices in WLANs has introduced significant security challenges, as the resource constraints and inadequate security mechanisms of these devices render them vulnerable to cyberattacks. Unidentified IoT devices can serve as entry points for broader network attacks or participate in malicious traffic generation, making their reliable identification a critical prerequisite for implementing targeted security measures such as network segmentation. Because modern wireless network traffic is typically encrypted, traditional payload-based analysis methods are ineffective for device detection. Furthermore, existing identification methods mostly rely on active network analysis, requiring IoT devices to maintain active network connections and be associated with an AP. The primary objective of this thesis was to address this limitation by developing a machine-learning method to reliably distinguish IoT devices from other network traffic based on the passive observation of MAC layer characteristics in individual IEEE 802.11 WLAN frames, enabling device identification even when the targeted devices have not yet established a network connection.

To address the identified security challenges, this thesis studied two primary questions. The first question examined how reliably IoT devices can be identified in dynamic IEEE 802.11 WLAN traffic using machine learning models trained on features extracted from individual MAC frames captured passively in monitor mode. The second question investigated what machine learning architecture is most

suitable for processing structural and dynamic single-frame features for IoT device identification, and what level of classification performance can be achieved. To answer the established research questions, a methodology grounded in passive network traffic analysis was implemented. Network traffic data used in the thesis was collected using monitor mode to capture MAC layer characteristics without requiring active device association. During the data preprocessing phase, raw network traffic was isolated into individual IEEE 802.11 WLAN frames, represented as fixed 49-byte sequential vectors, and augmented with dynamic traffic metadata, specifically inter-arrival times and frame lengths. A dual-branch fusion architecture, combining a 1D-CNN for the structural byte sequences and a MLP for the traffic metadata, was developed to perform the binary classification task. Finally, the operational reliability and classification accuracy of the developed fusion model were evaluated using a device-level group-aware cross-validation protocol, probability calibration, and specific operational threshold selection on an isolated test set.

To address the first research question, the results demonstrate that IoT devices can be reliably identified from dynamic IEEE 802.11 WLAN traffic using machine learning models trained on passively captured, individual MAC frames. By bypassing payload decryption and session tracking, the dual-branch fusion method achieves an operational precision of 99.5 % and a recall of 83.1 % on completely unseen devices. The achieved 99.5 % precision confirms the operational reliability of the method for network security applications, as the high precision effectively minimizes the false positive alarms generated by regular user devices. However, the evaluation also reveals that the standardized nature of the IEEE 802.11 standard imposes a natural upper limit on identification reliability during single-frame analysis. Because devices intended for distinctly different purposes frequently utilize identical header structures, such as QoS Data frames for traffic prioritization, the structural ambiguity prevents perfect classification accuracy. Despite the protocol-

level limitation, the achieved classification accuracy provides a functional technical foundation for automated network access management.

To answer the second research question, the evaluation confirms that a dual-branch fusion architecture is the most suitable machine learning approach for processing both static protocol structures and dynamic traffic behavior at the single-frame level. While ablation experiments showed that models relying solely on raw byte data struggle to detect a sufficient proportion of IoT traffic, and models based exclusively on dynamic metadata generate an unacceptable number of false alarms, the dual-branch fusion model balances the classification trade-offs to achieve a 90.5 % F1-score. The necessity of the dual-branch fusion architecture is particularly evident when confronting feature shift in dynamic wireless environments. When the temporal rhythm and packet sizes of IoT traffic and non-IoT traffic overlap in previously unseen networks, machine learning models dependent entirely on metadata lose their discriminative power. The dual-branch fusion model tolerates the feature shift by anchoring classification decisions to the underlying protocol structures processed by the 1D-CNN, ensuring that classification performance remains stable even when the network behavior of newly connected devices varies. Integrating both data modalities is therefore a highly effective architectural solution for maintaining robust device identification across changing network conditions.

The results confirm that the developed method supports automated network access management. By identifying devices without active connections or payload decryption, the dual-branch architecture provides a technical foundation for network segmentation. However, identification accuracy at the single-frame level is fundamentally constrained by the structural ambiguity of the IEEE 802.11 WLAN standard, as different device classes could share identical header fields. Furthermore, passive monitoring cannot detect physical sensors operating behind gateways. Future research should address these limitations by evaluating the computational

scalability of the model and integrating semi-supervised learning techniques to improve device profiling. The proposed architecture demonstrates high potential for cybersecurity applications, although the broad generalizability of the results to real-world networks requires further development and validation with significantly more diverse network traffic datasets.

References

[1] M. Abdur, S. Habib, M. Ali, and S. Ullah, "Security issues in the internet of things (iot): A comprehensive study", *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 6, 2017. DOI: 10.14569/IJACSA.2017.080650.

[2] T. Sasi, A. H. Lashkari, R. Lu, P. Xiong, and S. Iqbal, "A comprehensive survey on iot attacks: Taxonomy, detection mechanisms and challenges", *Journal of Information and Intelligence*, vol. 2, no. 6, pp. 455–513, 2024. DOI: 10.1016/j.jiixd.2023.12.001.

[3] G. Kambourakis, C. Kolias, and A. Stavrou, "The mirai botnet and the iot zombie armies", in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, Baltimore, MD, USA, 2017, pp. 267–272. DOI: 10.1109/MILCOM.2017.8170867.

[4] Sivanathan, Arunan and Gharakheili, Hassan Habibi and Loi, Franco and Radford, Adam and Wijenayake, Chamith and Vishwanath, Arun and Sivaraman, Vijay, "Classifying iot devices in smart environments using network traffic characteristics", *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2019. DOI: 10.1109/TMC.2018.2866249.

[5] N. Yousefnezhad, A. Malhi, and K. Främling, "Automated iot device identification based on full packet information using real-time network traffic", *Sensors*, vol. 21, no. 88, p. 2660, 2021. DOI: 10.3390/s21082660.

[6] M. R. P. Santos, R. M. C. Andrade, D. G. Gomes, and A. C. Callado, "An efficient approach for device identification and traffic classification in iot ecosystems", in *2018 IEEE Symposium on Computers and Communications (ISCC)*, Natal, Brazil, 2018, pp. 00 304–00 309. DOI: 10.1109/ISCC.2018.8538630.

[7] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "Audi: Toward autonomous iot device-type identification using periodic communication", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402–1412, 2019. DOI: 10.1109/JSAC.2019.2904364.

[8] A. J. Pinheiro, J. de M. Bezerra, C. A. P. Burgardt, and D. R. Campelo, "Identifying iot devices and events based on packet length from encrypted traffic", *Computer Communications*, vol. 144, pp. 8–17, 2019. DOI: 10.1016/j.comcom.2019.05.012.

[9] P. Sethi and S. R. Sarangi, "Internet of things: Architectures, protocols, and applications", *Journal of Electrical and Computer Engineering*, vol. 2017, no. 1, p. 9 324 035, 2017. DOI: 10.1155/2017/9324035.

[10] S. Alahmadi, P. Rojas, H. Idriss, and M. Bayoumi, "Taxonomy of consumer and industrial iot", in *SoutheastCon 2023*, Orlando, FL, USA, 2023, pp. 418–424. DOI: 10.1109/SoutheastCon51012.2023.10115217.

[11] J. Wurm, K. Hoang, O. Arias, A.-R. Sadeghi, and Y. Jin, "Security analysis on consumer and industrial iot devices", in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, Macao, China, 2016, pp. 519–524. DOI: 10.1109/ASPDAC.2016.7428064.

[12] A. Abdrabou, M. S. A. Darei, M. Prakash, and W. Zhuang, "Application-oriented traffic modeling of wifi-based internet of things gateways", *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1159–1170, 2022. DOI: 10.1109/JIOT.2021.3079115.

[13] T. Garrett, S. Dustdar, L. C. E. Bona, and E. P. Duarte, "Traffic differentiation on internet of things", in *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, Bamberg, Germany, 2018, pp. 142–151. DOI: 10.1109/SOSE.2018.00026.

[14] K. Zia, A. Chiumento, and P. Havinga, "Optimizing qos in wireless iot networks: A cross-layer based experimental study", in *2024 IEEE 29th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, Athens, Greece, 2024, pp. 1–7. DOI: 10.1109/CAMAD62243.2024.10942722.

[15] J. Zhang, H. Chen, L. Gong, J. Cao, and Z. Gu, "The current research of iot security", in *2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC)*, Hangzhou, China, 2019, pp. 346–353. DOI: 10.1109/DSC.2019.00059.

[16] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, "Iot: Internet of threats? a survey of practical security vulnerabilities in real iot devices", *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182–8201, 2019. DOI: 10.1109/JIOT.2019.2935189.

[17] T. Bakhshi, B. Ghita, and I. Kuzminykh, "A review of iot firmware vulnerabilities and auditing techniques", *Sensors*, vol. 24, no. 22, p. 708, 2024. DOI: 10.3390/s24020708.

[18] C. Bellman and P. C. v. Oorschot, "Best practices for iot security: What does that even mean?", 2020. DOI: 10.48550/arXiv.2004.12179.

[19] M. Hyppönen, *If It's Smart, It's Vulnerable*. Newark, UNITED STATES: John Wiley Sons, Incorporated, 2022, ISBN: 978-1-119-89519-0.

[20] K. Pahlavan and P. Krishnamurthy, "Evolution and impact of wi-fi technology and applications: A historical perspective", *International Journal of Wireless*

Information Networks, vol. 28, no. 1, pp. 3–19, 2021. DOI: 10.1007/s10776-020-00501-8.

[21] A. Garcia-Rodriguez, D. López-Pérez, L. Galati-Giordano, and G. Geraci, "Ieee 802.11be: Wi-fi 7 strikes back", *IEEE Communications Magazine*, vol. 59, no. 4, pp. 102–108, 2021. DOI: 10.1109/MCOM.001.2000711.

[22] G. H. Geleta, D. M. Molla, and K. A. Fante, "Comparative study of modulation techniques for 5g networks", in *Advances of Science and Technology*, F. A. Zimale, T. Enku Nigussie, and S. W. Fanta, Eds. Springer International Publishing, 2019, vol. 274, pp. 503–518. DOI: 10.1007/978-3-030-15357-1_41.

[23] N. K. Ojha and E. Baray, "An overview of protocols-based security threats and countermeasures in wlan", in *2023 4th International Conference for Emerging Technology (INCET)*, Belgaum, India, 2023, pp. 1–6. DOI: 10.1109/INCET57972.2023.10170514.

[24] M. Vanhoef and J. Robben, "A security analysis of wpa3-pk: Implementation and precomputation attacks", in *Applied Cryptography and Network Security*, C. Pöpper and L. Batina, Eds., Berlin, Heidelberg: Springer Nature Switzerland, 2024, pp. 217–240. DOI: 10.1007/978-3-031-54773-7_9.

[25] E. Baray and N. Kumar Ojha, "'wlan security protocols and wpa3 security approach measurement through aircrack-ng technique'", in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, 2021, pp. 23–30. DOI: 10.1109/ICCMC51019.2021.9418230.

[26] V. M. G. Martínez, M. R. N. Ribeiro, and V. F. S. Mota, "Wi-fi faces the new wireless ecosystem: A critical review", *Annals of Telecommunications*, vol. 79, no. 5, pp. 397–413, 2024. DOI: 10.1007/s12243-023-00995-2.

[27] J. Wang, G. Huang, and Z. Shao, "Performance evaluation of wi-fi 6 and technology prospects of wi-fi", in *2022 International Conference on Information Processing and Network Provisioning (ICIPNP)*, Beijing, China, 2022, pp. 91–95. DOI: 10.1109/ICIPNP57450.2022.00026.

[28] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset", *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 184–208, 2016. DOI: 10.1109/COMST.2015.2402161.

[29] A. Paramonov, A. Vikulov, and S. Scherbakov, "Practical results of wlan traffic analysis", in *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*, O. Galinina, S. Andreev, S. Balandin, and Y. Koucheryavy, Eds., Cham: Springer International Publishing, 2017, pp. 721–733.

[30] B. Alipour, L. Tonetto, A. Y. Ding, R. Ketabi, J. Ott, and A. Helmy, "Flutes vs. cellos: Analyzing mobility-traffic correlations in large wlan traces", in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Honolulu, HI, USA, 2018, pp. 1637–1645. DOI: 10.1109/INFOCOM.2018.8486360.

[31] R. R. Chowdhury, A. C. Idris, and P. E. Abas, "Packet-level and ieee 802.11 mac frame-level analysis for iot device identification", *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 30, no. 5, pp. 1941–1961, 2022. DOI: 10.55730/1300-0632.3915.

[32] X. Liu, Y. Han, and Y. Du, "Iot device identification using directional packet length sequences and 1d-cnn", *Sensors*, vol. 22, no. 2121, p. 8337, 2022. DOI: 10.3390/s22218337.

[33] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic", in *ICC 2019 - 2019 IEEE International Conference on Commu-*

nications (ICC), Shanghai, China, 2019, pp. 1–7. DOI: 10.1109/ICC.2019.8761559.

[34] G. Bendiab, S. Shiaeles, A. Alruban, and N. Kolokotronis, "Iot malware network traffic classification using visual representation and deep learning", in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, Ghent, Belgium, 2020, pp. 444–449. DOI: 10.1109/NetSoft48620.2020.9165381.

[35] B. Hartpence, "Neural network architectures and ensembles for packet classification: Addressing visibility, security and quality of service challenges in communication networks", PhD dissertation, Rochester Institute of Technology, Rochester, NY, USA, 2020. [Online]. Available: https://repository.rit.edu/theses/10486/.

[36] C. Duan, H. Gao, G. Song, J. Yang, and Z. Wang, "Byteiot: A practical iot device identification system based on packet length distribution", *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1717–1728, 2022. DOI: 10.1109/TNSM.2021.3130312.

[37] N. Alqudah and Q. Yaseen, "Machine learning for traffic analysis: A review", *Procedia Computer Science*, vol. 170, pp. 911–916, 2020. DOI: 10.1016/j.procs.2020.03.111.

[38] F. Hu, S. Zhang, X. Lin, L. Wu, N. Liao, and Y. Song, "Network traffic classification model based on attention mechanism and spatiotemporal features", *EURASIP Journal on Information Security*, vol. 2023, no. 1, p. 6, 2023. DOI: 10.1186/s13635-023-00141-4.

[39] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning", *Electronic Markets*, vol. 31, no. 3, pp. 685–695, 2021. DOI: 10.1007/s12525-021-00475-2.

[40] N. Ammar, L. Noirie, and S. Tixeuil, "Autonomous identification of iot device types based on a supervised classification", in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, 2020, pp. 1–6. DOI: 10.1109/ICC40277.2020.9148821.

[41] J. D. Yoo, Y. H. Lee, and H. K. Kim, *Iot-environment-dataset*, 2020. [Online]. Available: https://ocslab.hksecurity.net/Datasets/iot-environment-dataset.

[42] R. Roy Chowdhury, S. Aneja, N. Aneja, and P. E. Abas, "Packet-level and ieee 802.11 mac frame-level network traffic traces data of the d-link iot devices", *Data in Brief*, vol. 37, p. 107 208, 2021. DOI: 10.1016/j.dib.2021.107208.

[43] A. Holgado Moreno, M. J. López Salmerón, and L. Redondo Lopez, *Encrypted network traffic analysis: Iot/non-iot and cyberattack dataset.* 2025. DOI: 10.5281/zenodo.14802737.

[44] L. Pintor, A. Chirigu, and L. Atzori, *A dataset of labelled device wi-fi probe requests for mac address de-randomization - 2024*, 2025. DOI: 10.17632/5tvnwhsj2p.2.

[45] T. Baltrusaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 423–443, 2019. DOI: 10.1109/TPAMI.2018.2798607.

[46] N. Bayat, W. Jackson, and D. Liu, "Deep learning for network traffic classification", no. arXiv:2106.12693, 2021. DOI: 10.48550/arXiv.2106.12693.

[47] Sivanathan, Arunan and Sherratt, Daniel and Gharakheili, Hassan Habibi and Radford, Adam and Wijenayake, Chamith and Vishwanath, Arun and Sivaraman, Vijay, "Characterizing and classifying iot traffic in smart cities and campuses", in *2017 IEEE Conference on Computer Communications Work-*

shops (INFOCOM WKSHPS), Atlanta, GA, USA, 2017, pp. 559–564. DOI: 10.1109/INFCOMW.2017.8116438.

[48] J. T. Hancock, T. M. Khoshgoftaar, and J. M. Johnson, "Evaluating classifier performance with highly imbalanced big data", *Journal of Big Data*, vol. 10, no. 1, p. 42, 2023. DOI: 10.1186/s40537-023-00724-5.

[49] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks", in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, Sydney, NSW, Australia: JMLR.org, 2017, pp. 1321–1330.

[50] C. L. Calvert and T. M. Khoshgoftaar, "Threshold based optimization of performance metrics with severely imbalanced big security data", in *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, Portland, OR, USA, 2019, pp. 1328–1334. DOI: 10.1109/ICTAI.2019.00184.

[51] G. Tjhai, M. Papadaki, S. Furnell, and N. Clarke, "Investigating the problem of ids false alarms: An experimental study using snort", in *Proceedings of The Ifip Tc 11 23rd International Information Security Conference*, S. Jajodia, P. Samarati, and S. Cimato, Eds. Boston, MA: Springer US, 2008, vol. 278, pp. 253–267. DOI: 10.1007/978-0-387-09699-5_17.

[52] T. N. Ananna and M. Saifuzzaman, "Introduction to internet of things", in *IoT and ML for Information Management: A Smart Healthcare Perspective*, S. Namasudra, Ed. Springer Nature Singapore, 2024, pp. 1–49. DOI: 10.1007/978-981-97-5624-7_1.

[53] S. Balaji, K. Nathani, and R. Santhakumar, "Iot technology, applications and challenges: A contemporary survey", *Wireless Personal Communications*, vol. 108, no. 1, pp. 363–388, 2019. DOI: 10.1007/s11277-019-06407-w.

[54] I. Analytics, *State of iot 2024: Number of connected iot devices growing 13% to 18.8 billion globally*, Sep. 2024. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/.

[55] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey", *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010. DOI: 10.1016/j.comnet.2010.05.010.

[56] K. Zhang, X. Ge, C. Liu, and L. Xiang, "Analysis of frame traffic characteristics in ieee 802.11 networks", in *2009 Fourth International Conference on Communications and Networking in China*, Xi'an, China, 2009, pp. 1–5. DOI: 10.1109/CHINACOM.2009.5339870.

[57] R. R. Chowdhury, S. Aneja, N. Aneja, and E. Abas, "Network traffic analysis based iot device identification", in *Proceedings of the 2020 4th International Conference on Big Data and Internet of Things*, New York, NY, USA: Association for Computing Machinery, 2020, pp. 79–89. DOI: 10.1145/3421537.3421545.

[58] M. Kim, Z. Zhang, D. Kim, and S. Choi, "Deep-learning-based frame format detection for ieee 802.11 wireless local area networks", *Electronics*, vol. 9, no. 77, p. 1170, 2020. DOI: 10.3390/electronics9071170.

[59] H. F. Fakhruldeen, M. J. Saadh, S. Khan, N. A. Salim, N. Jhamat, and G. Mustafa, "Enhancing smart home device identification in wifi environments for futuristic smart networks-based iot", *International Journal of Data Science and Analytics*, 2024. DOI: 10.1007/s41060-023-00489-3.

[60] R. Chauhan, "A machine learning-based approach for network traffic analysis and management", *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 11, no. 33, pp. 2060–2066, 2020. DOI: 10.17762/turcomat.v11i3.13603.

[61] U. von Luxburg and B. Schölkopf, "Statistical learning theory: Models, concepts, and results", in *Inductive Logic*, D. M. Gabbay, S. Hartmann, and J. Woods, Eds., vol. 10, North-Holland, 2011, pp. 651–706. DOI: https://doi.org/10.1016/B978-0-444-52936-7.50016-1.

[62] M. Swarnkar, R. Kumar, R. Baidyo, and H. G, "Liteex: A lightweight feature extraction tool for captured network traces", in *2023 15th International Conference on COMmunication Systems NETworkS (COMSNETS)*, Bangalore, India, 2023, pp. 243–251. DOI: 10.1109/COMSNETS56262.2023.10041389.

[63] Y. H. L. Jeong Do Yoo and H. K. Kim, *Iot-environment-dataset*, 2020. [Online]. Available: https://ocslab.hksecurity.net/Datasets/iot-environment-dataset.

Appendix A Training Configuration

Component	Configuration
Training and validation	
Reproducibility	Random seed: 42.
Loss and metrics	Binary cross-entropy (label smoothing = 0.1); metrics: Accuracy, AUROC, AUPRC.
Optimizer	Adam; learning rate selected by tuning (Table 4.3).
Class imbalance handling	Class weights computed per fold using `class_weight="balanced"`.
Cross-validation protocol	Group-aware stratified CV (requested splits = 5; minimum allowed = 3; random state = 42). Validation constraints: minimum validation fraction = 0.05; minimum class fraction = 0.01.
CV training settings	Epochs = 12; batch size = 64; shuffle = true.
CV callbacks	EarlyStopping: monitor `val_auprc`, mode = max, patience = 3, restore best weights. ReduceLROnPlateau: monitor `val_loss`, mode = min, factor = 0.5, patience = 1.
Final training on full training data	Final epoch count set to the median of per-fold best epochs (minimum 3 epochs). Trained on the full training data set with batch size = 128 and class weights.
Decision-making	
Probability calibration	Platt scaling fitted on out-of-fold predictions using cross-fitted calibration (splits = 5; random state = 123).
Decision threshold selection	Threshold selected on calibrated OOF predictions via grid search (grid size = 801), maximizing F_β with $\beta = 2.0$ subject to an IoT recall constraint (recall ≥ 0.80).

Table A.1: Training, validation, and decision-making configuration for fusion model