Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# DIFT-VAR: A DYNAMIC MULTI-LAYER FRAMEWORK FOR DEVICE FINGERPRINTING OF IDENTICAL DEVICES

**MANOJ KUMAR VEMULA[1], KAILA SHAHU CHATRAPATI [2]**

[1]Research Scholar, Department of CSE, JNTUH, Hyderabad, India

[2]Professor, Department of CSE, JNTUH, Hyderabad, India

E-mail: [1]manojkumarv2251@gmail.com, [2]shahujntu@gmail.com

**ABSTRACT**

Device fingerprinting is a powerful technique for identifying devices in an IoT environment, offering multiple advantages such as enhanced security through device authentication, improved network management by monitoring device behaviors, and anomaly detection for identifying unauthorized or compromised devices. The majority of recent fingerprinting schemes consider a heterogeneous device environment and use different machine learning techniques to identify devices using network traffic, signal-level information, radio frequency characteristics, etc. However, fingerprinting devices of the same make and model is a significant challenge in modern IoT environments, where many devices often share identical hardware and software configurations. Existing techniques cannot reliably differentiate identical devices as they lack sufficient data. This paper proposes a novel approach for Device Identification and Fingerprinting with Time-Variant Adaptive Recognition (DIFT-VAR) based on multi-layer, time-varying feature extraction. We construct dynamic fingerprints that uniquely identify each device by monitoring and fusing features such as probe request behavior, clock skew, transport layer characteristics, and radio signal metrics over time. We utilize machine learning algorithms such as Random Forests to classify devices based on these dynamic fingerprints. We further propose the use of dynamic time warping (DTW) for feature alignment and classification. Experimental results demonstrate the efficacy of our approach in distinguishing identical devices with an accuracy of over 97% using standard machine learning metrics.

**Keywords:** *Device fingerprinting, IoT Security, Dynamic time warping (DTW), Time-variant feature extraction, Machine learning for IoT security.*

## 1. INTRODUCTION

An IoT (Internet of Things) device is a physical object embedded with sensors connected to the internet. This physical device differs from any typical computer whose primary functionality is not computing. IoT devices connected to the internet have many advantages, including convenience, comfort, safety, security, reliability, etc. The IoT market is a rapidly expanding industry that covers a wide range of applications across different business verticals, from retail to health, transport, manufacturing, entertainment etc. Ensuring the security of IoT devices is essential for their widespread adoption. Due to device constraints, traditional security solutions for conventional computing paradigms cannot be directly applied to IoT. Additional challenges include scalability, operating environments, diverse device architectures, platforms, and protocols within the IoT ecosystem. Notably, Statista projects that by the end of 2024, over 50 billion IoT devices will be connected to the internet. Protocols, architectures, and platforms employed by these devices vary greatly, and with a short product development life-cycle, the number of vendors manufacturing these products is numerous. These challenges and IoT's open operating environment pose several security challenges. Authenticating a device before providing it with network access is the first line of defense to protect against security attacks in the cyber world. Even though many cryptographic schemes tailored to the IoT environment have been developed, these schemes (except those that use PKI certificates) are vulnerable to node forgery or impersonation attacks, where the identity of the other legitimate devices is employed [1], with secret-keys (as security credentials) being the most popular way of authenticating a device, weak passwords followed by unpatched devices with reported security vulnerabilities subject IoT devices to compromise. Once compromised, these devices

can be exploited to leak confidential information, used as launch pads to launch large-scale attacks (for example, the recent Mirai Botnet), target critical infrastructure, etc. The risk of attacks is significantly heightened as devices are susceptible to hacking, compromising, and reverse engineering, as well as inadequate security management mechanisms in wireless network systems. Thus, multiple approaches are required to secure IoT devices, ranging from design considerations to monitoring for potential attacks and implementing effective mitigation strategies.

In a network, devices are usually identified using IP addresses, MAC addresses, device serial numbers, etc. The limitation of these identifiers is that they are susceptible to spoofing. The ability to spoof addresses of multiple legitimate devices allows the attacker to launch more sophisticated attacks without being detected. Recently, device fingerprinting has emerged as an alternative approach to identify devices, in which unique characteristics of a device are used to generate device signatures and used to identify devices [6]. The premise is to extract device features or to derive patterns through communications with the devices. Different network layers can contribute to this feature set, which can help in developing a fingerprint of the device. Device fingerprinting can be used not only to identify devices, but also to authenticate them, provided the fingerprint is unique [2]. Furthermore, a few works [3], [4], [5] have also employed fingerprinting to detect hidden eavesdroppers like a hidden camera, a network eavesdropper, or any such passive wireless devices to protect privacy. Also, identifying a device type helps in monitoring the behavior of a particular device, thereby differentiating the anomalous behavior from normal behavior. It also helps in isolating malicious nodes by maintaining an asset list. Unique fingerprints employed for authentication prevent the device identities from being forged.

The major steps involved in the fingerprinting process are: (1) identifying the relevant features, (2) extracting and modelling them, and (3) identifying the device. Employing suitable machine learning algorithms to achieve the above has made device fingerprinting an effective technique to address the unique security challenges posed by IoT networks.

The existing fingerprinting techniques can be categorized based on the features employed for identifying devices: for example, network-based fingerprinting techniques rely on network traffic patterns [4], Wi-Fi based techniques use medium access control (MAC) sub-layer information [5], clock-skew based methods use slight variations in clock of each device[8], methods based on electromagnetic emanations (EME) use unique signals from device components [7], and radio frequency (RF) based techniques use physical layer information like signal-to-noise ratio and other similar factors [8]. All these techniques rely on some type of machine learning algorithm to classify and identify devices.

However, the major challenge is to differentiate devices that are identical in terms of hardware and software configurations. As the majority of the IoT products are mass-manufactured, they tend to be of identical configuration, which makes the issue of device identification more challenging [9].

### 1.1. Motivation And Problem Statement

Typically, IoT devices deployed for a specific application, such as in smart homes, industrial monitoring, or healthcare environments, are comprised of devices that may be identical in terms of hardware and software configurations. The limitations of static network identifiers create a significant challenge in identifying and differentiating these devices once they are operational within the same network. Here, the task is to uniquely fingerprint IoT devices of the same make and model without relying on EME, which often requires specialized hardware and is impractical in large-scale deployments. Even though the devices are identical in both hardware and software, they still exhibit subtle differences in network behavior, communication timing, and usage patterns due to natural variations in manufacturing, environmental factors, and internal processing states. The majority of the existing works focused on the heterogeneous type of devices and have produced significant device profiling and authentication approaches. Moreover, we highlight the work by collecting minute differences in significant parameters with the network is in operation from the various homogeneous devices to address the authentication issues. In addition to this, the proposed model has produced tangible results in terms of fingerprinting identical devices. As the model focused on time series data, which is collected with variation in time slots.

This work aims to design a novel fingerprinting method to capture and analyze these subtle, naturally occurring variations in device behavior to uniquely identify each device. This fingerprinting method should integrate multi-layer features to identify and fingerprint devices.

### 1.2 Contributions

This paper makes the following contributions:

- We introduce a novel multi-layer approach to fingerprinting identical devices, leveraging time-varying features across the MAC, transport, and physical layers.
- A dynamic time warping (DTW)-based algorithm is proposed for temporal alignment and classification of device fingerprints.
- We demonstrate through extensive experiments that our approach can successfully differentiate devices of the same make and model, using real-world IoT data.

## 2. RELATED WORKS

Device identification and fingerprinting are key to improve the security posture in an IoT environment where devices are susceptible to compromise. In this paper, we present different works in this direction for identifying IoT devices, focusing on their effectiveness, technical details, strengths, and limitations. One of the simpler approaches to identifying devices is based on device signatures. These methods create rules based on known device behaviors, which can be effective for identification [10], [11]. However, these techniques require constant updates to adapt to new devices and threats. Alternatively, the network traffic generated by IoT devices can be used to learn unique patterns that can be used to classify and identify devices. Multiple such works use network traffic for device identification [12], [13], [14], [15], [16]. For example, the authors of [12] use machine learning-based methods to identify IoT devices through network traffic analysis from a test set of nine devices. The model achieves 99.28% accuracy in distinguishing devices utilizing network features. It performs session-level analysis and uses a multistage classifier for specific IoT identification.

Similarly, the work presented in [13] also uses different network traffic patterns such as types of traffic (e.g. different protocols), packet sizes, the frequency of packets along with their statistical measures, packet inter-arriving times, etc. to identify devices using machine learning algorithms and then apply predefined security measures following a security policy. In [14], the authors address the challenge of managing a large number of IoT devices in a large-scale IoT environment such as a smart city. The presented work uses network traffic analytics to characterize and monitor IoT device behavior by collecting traffic traces from a diverse set of IoT devices that include smart cameras, smart lights, and health monitors. Features such as data rates, activity cycles, and signaling patterns were employed to first distinguish IoT from non-IoT traffic and then identify specific IoT devices with more than 95% precision. Other works based on network characteristics, like [15], use genetic algorithms for feature selection and different machine learning algorithms for device classification. On the other hand, the authors in [16] address the problem of data imbalance in device classification. All of the above-mentioned approaches use different network characteristics and tailor their ML approaches for device classification and identification.

Other approaches for device identification are either RF-based, clock-skew based, or specialized hardware-based techniques that have their respective advantages and limitations. RF-based device identification schemes leverage the unique characteristics of the radio signals, such as signal's amplitude, frequency, and phase characteristics emitted by IoT devices, then apply ML algorithms to create a distinct fingerprint for each device. The authors of [17], [18] present a survey of such approaches. In [19], the authors propose to exploit the unique characteristics found in the energy spectrum of transmitter turn-on transient signals. These transient signals contain unique, hardware-specific variations that serve as fingerprints for device identification. The method extracts the energy distribution across different frequency components from these transients, which helps to distinguish between devices, even at low signal-to-noise ratios. In [20], the authors present an approach that converts the time series data into images instead of directly using raw signals or extracting statistical features from them. This transformation allows the use of well-established image-processing techniques and machine learning models that are particularly powerful for pattern recognition. Approaches based on clock skew take advantage of minute differences in the internal clock rates of devices to create unique identifiers [21], [8]. This is done by monitoring the network packets sent by devices over time to calculate the deviation of each device's clock from a reference clock. Then, a unique profile is created based on the consistent skew patterns observed. A survey of such approaches is presented in [18]. More recently, a clock skew-based device identification scheme combined with a remote attestation protocol has been presented in [22] for class-I IoT devices (devices with random access memory less than 10 KB and code size less than 100 KB). The major challenges with the clock skew-based

approaches are that they need continuous monitoring over an extended period of time and are affected by network latency and jitter.

Apart from the aforementioned device identification approaches, few other works consider a hybrid approach [23], that is to combine multiple device characteristics and other uses device sensors for fingerprinting [24]. However, none of the above approaches consider multiple identical devices with the same hardware and firmware in their test setup and evaluate the device identification and fingerprinting approaches. In this work, we propose a novel approach for Device Identification and Fingerprinting with Time-Variant Adaptive Recognition (DIFT-VAR) that is based on multi-layer, time-varying feature extraction. we construct dynamic fingerprints that uniquely identify each device by monitoring and fusing features such as probe request behavior, clock skew, transport layer characteristics, and radio signal metrics over time. Alternatively, other works have explored the issue of fingerprinting in a heterogeneous environment. For example, the work presented in [25] presents a method to classify IoT device types (e.g., cameras, routers, printers) using Shodan metadata. It emphasizes on the effectiveness of using curated metadata and machine learning for accurate identification of IoT device types. The paper [26] explores the use of ICMP and IP timestamp responses to distinguish between physical Android devices and virtual machines. Designed to detect VM-based malware evasion, the study reveals consistent timing discrepancies that serve as passive indicators of the execution environment. The paper in [27] employs fingerprints for IOT devices. It is network behavior-based, extracting features from the network, Transport, and application layers, and has also employed neural networks for classification. Employed neural network to classify IoT devices by utilizing 3 stages, i.e, device type, vendor, and product. He has employed a glove and a Bi-LSTM for the application layer data. Device behavior, however, varies as when firmware updates. The paper [28] utilizes the MRFE deep learning approach to recognize IoT devices using RF fingerprints. The model improves the identification accuracy with multi-dimensional features. The data set was acquired in noise noise-free environment. The model uses a fingerprint-amplifying layer, three-channel input, an Attention mechanism, and residual connections and fully connected layers. The model depends more on the behavior of the RF data. However, the performance of the model drops to below 8dB of SNR

## 3. DIFT-VAR: THE PROPOSED FINGERPRINTING TECHNIQUE

Device identification in a heterogeneous IoT environment is comparatively simpler as the devices exhibit sufficiently varied characteristics that can be captured via network, RF, device behavior, or clock skew, and other similar approaches. However, in an IoT environment where multiple devices share the same device architecture in terms of both hardware and firmware, the issue of device identification becomes a challenge. The interaction of these devices in a network exhibits almost similar patterns that are hard to distinguish, especially when the devices are operational on the same application. A typical use-case scenario is a surveillance system where a set of smart security cameras can belong to a single manufacturer. The surveillance application necessitates that all devices report surveillance data that can exhibit similar communication patterns. In such a scenario, the key is to exploit subtle variations in their behavior that arise from hardware imperfections, network conditions, and environmental factors. In this work, we propose a novel method of processing the data by leveraging multi-layer data fusion combined with time-varying feature extraction to create unique device fingerprints over time. The core idea is to design a multi-layer, time-varying fingerprint Construction method. Rather than relying on a static feature set, the proposed approach focuses on the temporal evolution of features across multiple layers (MAC layer, radio characteristics, transport layer, and timing data) to differentiate devices. The novelty is in dynamically combining temporal trends and subtle variations in network and timing behavior over time, rather than static snapshots of features.

The proposed approach is named DIFT-VAR, where a time-varying multi-layer fingerprinting approach that fuses features across the MAC, transport, and timing layers. Our approach consists of three main components: multi-layer feature extraction, time-varying feature aggregation, and classification using dynamic time warping (DTW). The uniqueness lies in the processing method, specifically in differentiating identical devices based on a temporal dependency analysis using dynamic time warping (DTW) and recurrent neural networks (RNN) with an emphasis on sequence processing.

### 3.1 Multi-Layer Feature Fusion and Temporal Variation Tracking.

As identical devices may exhibit slight hardware-induced variations over time due to environmental factors (like distance from the Wi-Fi access point), small variations in the internal clock, or processing speed differences that arise from manufacturing imperfections, we do not rely on a fixed dataset of features collected at one point in time. We consider the evolution of features over time and fuse data across multiple layers (MAC, radio, transport, and timing) to create a highly unique fingerprint.

At the MAC layer, we consider the MAC address and the received signal strength indicator (RSSI) of probe requests. The MAC address is considered a basic identifier. Even though it can easily be spoofed, it is considered a starting point. We consider the Wi-Fi communication among devices where devices periodically send probe requests for network discovery and re-association. Let $T_i^{(k)}$ represent the interval between the *i-th* and *(i+1)-th* probe requests of the device $k$. To compare devices, we construct a time-series vector and use Dynamic Time Warping (DTW) to calculate the similarity between two time series. $(S(\mathbf{T}^{(k)}, \mathbf{T}^{(m)})$:

$$DTW(\mathbf{T}^{(k)}, \mathbf{T}^{(m)}) = \min \sum_{i=1}^{n} d(T_i^{(k)}, T_j^{(m)})$$

where $d(\cdot,\cdot)$ It is a distance metric (typically Euclidean distance).

The time intervals between consecutive probe requests can vary slightly due to internal hardware differences and the timing precision of each device. The probe requests are analyzed for supported data rates, SSID, and capabilities. Even identical devices can have subtle differences based on firmware or environmental adaptations. The timing of probe requests is used to generate a temporal pattern for each device, which is then compared using a dynamic time-warping approach to identify subtle differences in request intervals.

The RSSI values are captured for packets between the devices and the Wi-Fi access point. These values are collected continuously since physical placement and antenna differences cause variations. For signal-to-noise ratio (SNR), the signal quality relative to the background noise is measured. The differences in the device's RF components can result in slight but detectable variations in SNR. Also, as each device may have slight variations in transmission power, it can lead to differences in RSSI over time. We set up multiple access points to measure how frequently the devices switch channels when numerous access points are available, considering their channel utilization capabilities.

Let $\mathbf{RSSI}^{(k)} = [r_1^{(k)}, r_2^{(k)}, \ldots, r_n^{(k)}]$ be the RSSI readings of the device $k$. For differentiation, calculate the variance over a sliding window of size $w$:

$$\sigma_{RSSI}^{(k)} = \frac{1}{w}\sum_{i=1}^{w}(r_i^{(k)} - \mu_{RSSI})^2, \quad \text{where } \mu_{RSSI} = \frac{1}{w}\sum_{i=1}^{w} r_i^{(k)}$$

To capture changes over time, generate an aggregated feature vector for each time window:

$$\mathbf{F}_{RSSI}^{(k)} = [\sigma_{RSSI}^{(1)}, \sigma_{RSSI}^{(2)}, \ldots, \sigma_{RSSI}^{(m)}]$$

The RSSI and SNR are collected as time-series data. This information is processed using sequence models to identify the drift in signal strength or noise conditions, providing unique identifiers for each device regarding their physical connectivity to the network.

We consider the TCP window size at the transport layer level and monitor the changes during data exchanges. TCP sequence numbers are observed during the handshake and communication phases. The difference between sequence numbers, or the sequence number gaps, can vary depending on how the device's network stack handles retransmissions and acknowledgments.

Let the TCP sequence numbers be represented as

$$\mathbf{SEQ}^{(k)} = [seq_1^{(k)}, seq_2^{(k)}, \ldots, seq_n^{(k)}]$$

Using DTW for alignment of sequence numbers to identify subtle device-specific variations, it is represented as follows:

$$DTW(\mathbf{SEQ}^{(k)}, \mathbf{SEQ}^{(m)}) = \min_{\text{paths}} \sum_{i=1}^{n} \| seq_i^{(k)} - seq_j^{(m)} \|$$

We consider the round-trip time (RTT) for TCP connections and consistently measure the RTT between the devices and a server on the network. Its value varies based on clock skew and processing speed differences. The TCP sequence number and window size are modeled as features contributing to network behavior differences. The RTT is treated as temporal data and is continuously monitored to build a clock skew profile over time.

The RTT is represented as $RTT_i^{(k)}$ for device k . Using the time-averaged mean and variance:

$$\mu_{RTT}^{(k)} = \frac{1}{n}\sum_{i=1}^{n} RTT_i^{(k)}, \quad \sigma_{RTT}^{(k)} = \frac{1}{n}\sum_{i=1}^{n} (RTT_i^{(k)} - \mu_{RTT}^{(k)})^2$$

Clock skew $C(t)$ is the deviation of the clock over time. The clock skew is measured by sending ICMP pings to each device and recording the round-trip time (RTT). The drift over time is calculated to produce a clock skew value that is unique to each device. On the other hand, the inter-arrival time (IAT) for packets, particularly during high activity, is measured. The clock skew function for the device $k$ can be represented as:

$$C_k(t) = \alpha_k t + \beta_k$$

where $\alpha_k$ is the drift rate and $\beta_k$ represents the initial offset. Comparing clock skews of two devices involves computing the difference:

$$\Delta C_{k,m}(t) = | C_k(t) - C_m(t) |$$

Differences in clock drift and packet processing speed result in detectable variations in packet timing. The clock skew and IAT are analyzed as time-series features, providing a unique temporal signature for each device.

### 3.2 Time-Varying Feature Aggregation

The proposed approach, algorithm 1, considers features collected over time rather than static features collected at a single time instance. This enables us to capture temporal patterns in network behavior that are subtle but consistent across identical devices. We show that time-varying patterns often reveal minor differences due to environmental or hardware factors. To achieve that, we convert key features into time series data and apply time series analysis techniques like moving averages, Fourier transforms, and wavelet analysis to capture temporal trends in feature behavior. The method chosen is dependent on the feature type. For example, to analyze the RSSI values that fluctuate due to environmental conditions, we consider moving averages to smooth out the signal strength variations.

On the other hand, to analyze probe request intervals and suspect that a device sends requests at regular intervals, we apply a Fourier transform that will highlight this cyclical pattern. The Fourier spectrum will peak at the corresponding frequency, thus allowing the differentiation of devices based on their periodic behavior. For features such as clock skew that exhibit a long-term drift spread across with occasional variations due to internal processing delays, we employ wavelet analysis that allows us to capture both the gradual changes and the sudden shifts in timing behavior.

**Algorithm-1** Multi-Layer Device Fingerprinting with Temporal Sequence Analysis

1. **Input:**
2. Set of $N$ devices $\{D_1, D_2, \ldots, D_N\}$
3. For each device $D_i$ , collect time-series data $X_i$ :
4. MAC layer features $MAC_i$
5. Radio characteristics $RSSI_i, SNR_i$
6. Transport layer features $TCPSeq_i, RTT_i, TCPWnd_i$
7. Timing features $Probe\ Interval_i, Clock\ Skew_i$
8. **Feature Extraction:**
9. For each device $D_i$ , extract time-series features:
10. MAC layer features: $MAC_i, Probe\ Interval_i$
11. Radio characteristics: $RSSI_i, SNR_i$
12. Transport layer features: $TCPSeq_i, RTT_i, TCPWnd_i$
13. Timing features: $Clock\ Skew_i$
14. **Time-Series Alignment Using DTW:**
15. For each device $D_i$ and feature $X_i$ , perform DTW on sequences of:
16. $Probe\ Interval_i$ , $RTT_i$
17. Compute the DTW distance to align timing variations:
18. $S_{DTW}(X_i)$
19. **Temporal Modeling Using LSTM:**
20. Train an LSTM on time-series data $Xi$ for each feature:
21. $RSSI_i, RTT_i, TCPSeq_i, Clock\ Skew_i$
22. Output a sequence embedding $E_{LSTM}(X_i)$ for each device
23. **Feature Fusion:**
24. Fuse all features into a feature vector $Fi$ :

$$F_i = \begin{Bmatrix} MAC_i, RSSI_i, SNR_i, \\ S_{DTW}(X_i), E_{LSTM}(X_i), \\ TCPSeq_i, RTT_i, TCPWnd_i \end{Bmatrix}$$

**25. Classification:**

26. Train a Random Forest classifier using the feature vectors $F_i$
27. **Authentication:**
28. Given an unknown device $D_j$ , extract the feature vector $F_j$
29. Classify $F_j$ using the trained classifier to authenticate $D_j$ .
30. **Output:** Device classification and unique fingerprint for authentication.

Further, we employ a sliding window approach where the key features are calculated dynamically within time windows rather than static feature vectors, thus creating a temporal fingerprint with dynamic windows. For a time-varying feature $X^{(k)}(t)$ , we extract features using a sliding window approach of size $w$ .

$$X^{(k)}_{window}(t) = \frac{1}{w}\sum_{i=t}^{t+w} X^{(k)}(i)$$

For each time window, we capture and summarize variations in key features like clock skew, RSSI, TCP behavior, and probe request timing. For each of the device, the aggregate features from multiple layers within a dynamic time window creates a composite fingerprint. We employ a weighted approach to give more importance to layers that exhibit greater variability between devices. Features calculated for each sliding window are used to generate dynamic fingerprints that emphasize variability over time. The key components of the proposed architecture are shown in Figure 1.

### 3.3 Multi-Resolution Feature Matching

To uniquely differentiate devices, the proposed approach performs a multi-resolution analysis, where features are compared at different levels of granularity. To begin with, for each of the devices, we create both short-term and long-term feature profiles. The short-term profiles capture immediate fluctuations, while long-term profiles track cumulative behavior over an extended period of time. Then, we aim to differentiate devices based on a combination of short-term and long-term profiles using techniques like dynamic time warping (DTW) to align and compare time series data, accounting for slight time shifts or phase differences.

We define the short-term $\mathbf{S}^{(k)}$ and long-term $\mathbf{L}^{(k)}$ feature vectors for each device $k$ as follows:

$$\mathbf{S}^{(k)} = [s_1^{(k)}, s_2^{(k)}, \ldots, s_n^{(k)}], \quad \mathbf{L}^{(k)} = [l_1^{(k)}, l_2^{(k)}, \ldots, l_m^{(k)}]$$

DTW is used to measure the similarity between temporal sequences of different devices. DTW allows for alignment of time-series data that might be out of phase but still represent similar patterns. We apply DTW to align and compare the device profiles using:

$$DTW(\mathbf{S}^{(k)}, \mathbf{S}^{(m)}) + DTW(\mathbf{L}^{(k)}, \mathbf{L}^{(m)})$$

### 3.4 Feature Fusion and Classification

For differentiating and identifying devices we employ a feature fusion approach. In this method, we combine features from all layers (MAC, radio, transport, and timing) to create a comprehensive feature vector for each device. For any feature vector, let $\mathbf{F}^{(k)}$ represent the feature vector for device $k$, containing features from MAC, radio, transport, and timing layers:

$$\mathbf{F}^{(k)} = [F^{(k)}_{MAC}, F^{(k)}_{radio}, F^{(k)}_{transport}, F^{(k)}_{timing}, h^{(k)}_{LSTM}]$$

The weighted fusion of features, where weight $w_i$ corresponds to the variability of each feature is represented as:

$$F^{(k)}_{composite} = \sum_i w_i F_i^{(k)}, \quad with \sum_i w_i = 1$$

Next, we employ the Random Forest classifiers to train on the fused feature set. The LSTM output (temporal analysis) is also included in the feature set, providing additional temporal context to the classifier. The fused features are used to create a unique fingerprint for each device. The training of the Random Forest classifier on the fused feature vectors $\mathbf{F}^{(k)}$ is represented as follows, where $\hat{y}$ is the predicted class:

$$\hat{y} = Classifier(\mathbf{F}^{(k)})$$

The combined use of time-series alignment (DTW), temporal sequence modeling (LSTM), and layer-specific features makes the proposed approach unique.

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

### 3.5 Device Identification

The proposed approach identifies devices by collecting data over an extended period and creating a device profile. Continuous data collection involves continuously gathering features across all relevant layers to maintain both short-term and long-term profiles of each device. In the matching and classification process, DTW is applied to align newly collected time-series data with stored profiles. An LSTM model is used to infer temporal dependencies in real-time features. The aligned data and LSTM outputs are then fused and classified using a trained Random Forest classifier to authenticate the device. In the decision-making stage, authentication is determined based on the DTW alignment score and the Random Forest classification. It is represented as follows:

Device authentication is based on comparing the DTW alignment score $Score_{DTW}$ and classification outcome:

$$Authenticate\ if\ Score_{DTW} < \tau\ and\ \hat{y} = y_{true}$$

If the computed score is below the established threshold, the device is considered unrecognized or spoofed.

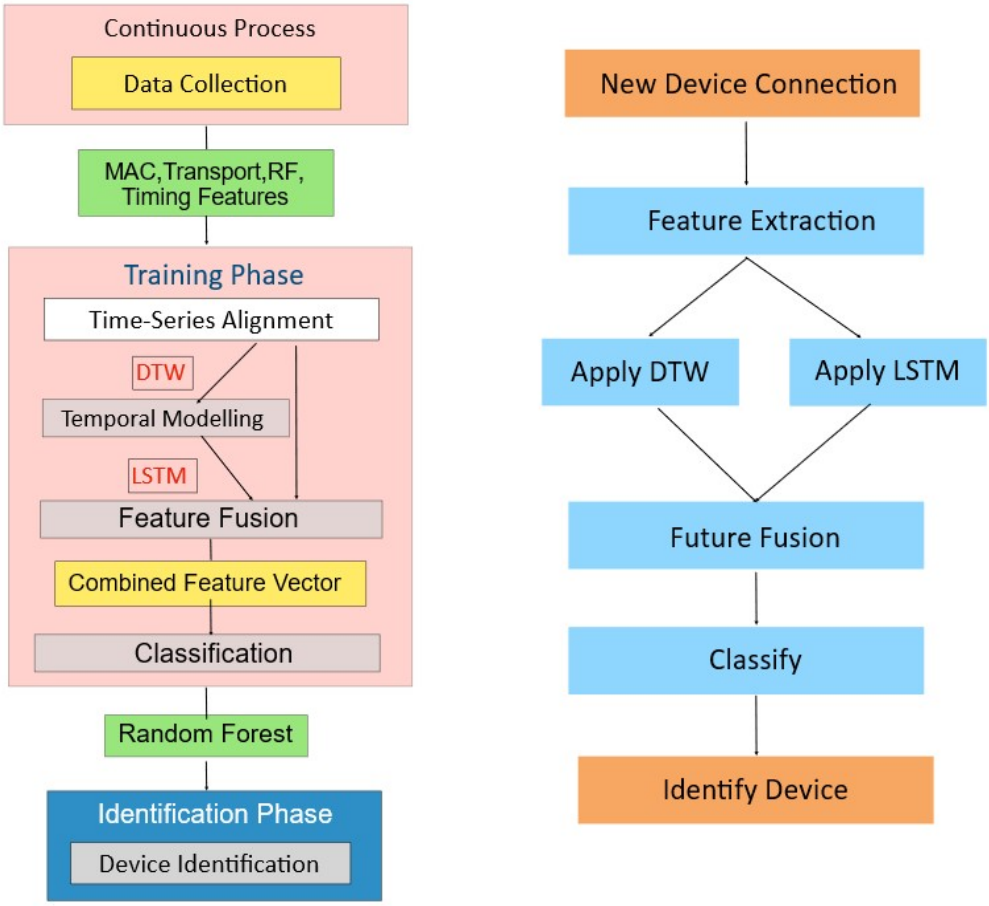


*Fig. 1 The proposed device identification model*

## 4. EXPERIMENTAL RESULTS

To analyze the performance of the proposed device identification and fingerprinting algorithm, we set up a network of 10 ESP32-WROOM-32 microcontrollers, all running the same firmware and communicating over a Wi-Fi network. Each device has the same set of sensors (temperature, humidity, light), which report data every 60 seconds. This experiment aims to evaluate the ability to uniquely fingerprint identical devices using multi-layer feature extraction and time-series processing techniques. The experiment will capture

the MAC layer, radio characteristics, transport layer, and timing features, then process the data using Dynamic Time Warping (DTW) and LSTM-based temporal modeling.

### 4.1 Hardware Set-up

Table 1 provides a detailed ESP32-WROOM-32 microcontroller specification summary. The boot process involves the first-stage boot loader, part of the ROM. The second-stage bootloader is stored in Flash and can be customized. The partition table defines the flash memory layout, and the application is part of the main firmware code. The ESP32 software development kit SDK, also known as the ESP-IDF (Espressif IoT Development Framework), provides a comprehensive set of tools, libraries, and application programming interfaces (APIs) that allow us to access and manipulate various hardware features of the ESP32. The SDK provides APIs for direct access to hardware features like ADC (Analog-to-Digital Converter), timers, and RF components. This low-level access allows for precise measurements of hardware characteristics. The ESP-IDF includes a full-featured Wi-Fi and Bluetooth stack, enabling fine-grained control over wireless communications. This can be used to implement custom transmission patterns or analyze reception characteristics. The SDK is built on FreeRTOS, enabling precise timing control and task management. It can be leveraged to create unique behavioral patterns or measure system response times accurately. Lastly, ESP-IDF provides advanced power management capabilities, allowing for measuring and controlling power consumption patterns, which can vary between devices.

The main CPU clock speed is up to 240 MHz with an internal 150 kHz RC oscillator and an external crystal of 40 MHz.. The internal oscillator is 8 MHz with calibration. The internal RC oscillator and PLL circuits can vary slightly due to manufacturing processes, affecting their exact frequencies. Also, temperature and voltage fluctuations can cause subtle clock drift patterns unique to each device. The ESP32's ability to dynamically adjust clock speeds can be used to create unique behavioral patterns. The device supports IEEE 802.11 b/g/n (2.4 GHz) protocols with an adjustable transmit power of up to +20 dBm for a receiver sensitivity of up to -98 dBm. All the devices are connected to the same Wi-Fi access point (TP-Link Archer C7). The devices use HTTP as the communication protocol to transmit sensor data.

*Table 1: Key Specifications of Esp32-Wroom-32.*

| Feature | Specification |
|---|---|
| Processor | Dual-core Xtensa LX6, 32-bit |
| Clock Speed | Up to 240 MHz |
| RAM | 520 KB SRAM |
| ROM | 448 KB |
| Flash Memory | 4 MB (expandable to 16 MB) |
| Wi-Fi | 802.11 b/g/n (2.4 GHz) |
| Bluetooth | v4.2 BR/EDR and BLE |
| GPIO Pins | 34 programmable |
| ADC | 12-bit, up to 18 channels |
| DAC | 2x 8-bit channels |
| Hardware Crypto | AES, SHA, RSA, ECC |
| RTC | 150 kHz internal oscillator |
| External Crystal | 40 MHz |
| Power Consumption | Average 80mA |
| Operating Voltage | 3.0V to 3.6V |
| Operating Temperature | -40°C to +85°C |

### 4.2 Data Collection and Data Processing

In this experiment, we collect data from 10 identical ESP32-WROOM-32 micro-controllers. Each device sends sensor data periodically. The following data is collected as shown in the Table.2

The features are collected over multiple sessions to ensure sufficient data is captured for analysis. The variations in the data are expected due to the minor hardware and environmental differences across devices, even though the devices are identical.

The data processing pipeline is comprised of two main stages, i.e., time-series alignment using DTW and temporal modeling using LSTM to capture the temporal dependencies of each of the device behavior. Time-series data for features like Probe Request Intervals, RTT, and IAT are extracted from each device. The aim of using DTW is to align sequences of time-series data to account for any variation due to differences in clock skew, processing delays, or environmental factors. For each feature (e.g., Probe Request Interval), we compute the DTW distance between each device's time-series data and a reference sequence. We then align the time-series data based on the minimum DTW distance to obtain a similarity score $(S_{DTW}(X_i)$ for each device $X_i$ . The outcome is a set of aligned sequences that capture the timing variations across devices.

To achieve temporal modeling using LSTM, we consider the time-series data from multiple sessions for features like RSSI, RTT, TCP sequence numbers, and clock skew as input. The LSTM network configuration has an input layer, where the input is the time-series data from each

device, with a hidden layer of 128 LSTM units that processes the input sequences. The LSTM outputs a sequence embedding $E_{LSTM}(X_i)$ , which is a low-dimensional representation of the temporal behavior of each device. All the features, including the DTW similarity scores and LSTM embeddings, are concatenated into a fused feature vector for each device, and this fused feature vector is used as the input for the classification model.

### 4.3 Classification and Fingerprinting

Once the feature vector space is formulated, the fused feature vectors train a Random Forest classifier to identify each device based on its unique features. Random Forest is well-suited for this task because it can efficiently process the multi-layered data (e.g., MAC layer, transport layer, timing data) while being robust to noise, overfitting, and feature interactions. This is critical when differentiating identical devices, where the signal between them may be subtle, and minor variations in multiple features must be captured.

The classification process helps determine the ability to differentiate between identical devices. We consider a Random Forest with 100 trees with fused feature vectors $F_i$ for each device. The predicted device ID for each feature vector is presented as the model's output. The Random Forest classifier is trained using 80% of the collected data, and the remaining 20% is used for testing. Typical metrics like accuracy, precision, recall, and F1 score are used to evaluate the classifier's performance. These metrics are generated to determine how well the classifier distinguishes between the identical ESP32 devices.

### 4.4 Results

The classification results show that the combination of time-series alignment, temporal modeling, and feature fusion yields high accuracy in fingerprinting identical devices. The Random Forest model provides feature importance scores as per the Table.3, indicating which features are most useful for differentiating the ESP32 devices.

The model achieved a device identification accuracy of 97.4% in identifying devices based on the collected feature set. Our first experiment is to measure the accuracy of the proposed device identification approach across multiple runs. The aim is to visualize the stability and consistency of the proposed model's performance across several trials of the experiment.

It can be observed from Figure 2 on how the model performs in each individual run and the variations in accuracy across different runs.

*Table.2: Data Collection for Esp32 Devices*

| Feature | Description | Data Generation |
|---|---|---|
| MAC Address | Unique identifier for each device | Each device gets a unique MAC address |
| Probe Request Interval | Time between consecutive probe requests | Random intervals between 50 and 100ms |
| RSSI | Signal strength of the device's Wi-Fi connection | RSSI values between -70 and -40 dBm |
| SNR | Signal-to-noise ratio of Wi-Fi connection | SNR values between 20 and 40 |
| TCP Sequence Number | Sequence number gaps during TCP connections | Random values between 1,000,000 and 1,001,000 |
| TCP Window Size | Size of the TCP window during communication | Random values between 64,000 and 65,535 Bytes |
| RTT | Round-trip time for data packets | RTT values between 50 and 150ms |
| Clock Skew | Drift in the device's internal clock over time | Skew values between 0.1ms and 0.5ms |
| IAT | Inter-packet arrival time | Random values between 5ms and 15ms |

The model consistently achieves accuracy around 96.5% to 97.3% across all runs, indicating that the classifier is stable and performs reliably in different test iterations. Even though there are minor variations in accuracy between the runs, the differences are insignificant. They are due to slight variations in network conditions or random training and testing data splits. However, the model reaches its highest accuracy in Run 2 (97.3%) and maintains similar high accuracy levels in Run 5 (97.2%), suggesting the model generalizes well and delivers consistent results.

The second experiment has been designed to analyze the performance of the proposed model in terms of other important metrics like precision, recall, and F1-score. Figure 3 presents the performance of the classification model across multiple runs, which helps in understanding the model behavior in terms of true positives, false positives, and false negatives.

*Table 3: Feature Importance Scores*

| Feature | Importance Score |
|---|---|
| MAC Address | 0.08 |
| Probe Request Interval | 0.12 |
| RSSI | 0.10 |
| SNR | 0.15 |
| TCP Sequence Number | 0.20 |
| TCP Window Size | 0.10 |
| RTT | 0.15 |
| Clock Skew | 0.10 |

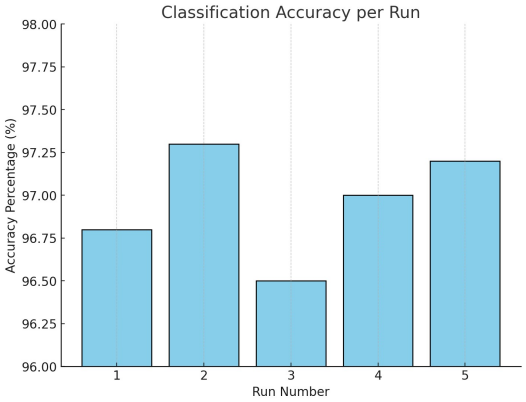


*Fig. 2. Classification Accuracy per Run*

Across all runs, precision, recall, and F1-score values are all very close, i.e., within the range of 96.4% to 97.3%. It indicates that the model is consistently effective across different aspects of classification (precision, recall).

In Run 2, the model performs the best with 97.0% precision, 97.3% recall, and 97.1% F1-score, indicating strong classification performance with a good balance between precision and recall.

The relatively high and balanced values for precision, recall, and F1-score suggest that the classifier can accurately predict with few false positives (high precision) and few missed true positives (high recall).

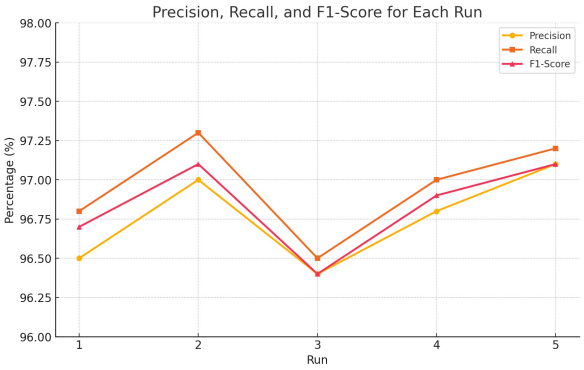


*Fig. 3. Precision, Recall, and F1 Score Comparison*

To evaluate the uniqueness of temporal communication patterns among identical IoT devices, we computed the pairwise DTW distances between them. By capturing features such as RTT, IAT, and clock skew, and aligning them using DTW, we show that minor variations in different features show subtle and consistent deviations in their behavior over time. we quantify these differences in the form of a distance matrix. The resulting DTW distance matrix visually represents the similarity between the temporal behavior of each device, where lower values indicate closely matching patterns and higher values suggest distinguishable behaviors. Figure 4 shows the DTW distance matrix for different devices. This approach provides a foundational rationale for using temporal features to uniquely fingerprint devices, even when they are otherwise indistinguishable by hardware or software alone. The observed separation in DTW values supports the effectiveness of time-series modeling as a reliable component of the proposed fingerprinting framework.

Finally, we present the confusion matrix for the proposed approach. It shows the number of times the model predicted a certain device (columns) when the actual device (rows) was being tested. The confusion matrix is shown in Figure 5.

Journal of Theoretical and Applied Information Technology
31st May 2025. Vol.103. No.10

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Table 4: Comparison of Proposed Work with Other Fingerprinting Approaches*

| Parameter | Proposed Work | Nazerian et al. [7] | Yang et al. [27] |
|---|---|---|---|
| Features Considered | Multi-layer features from MAC, Transport, Timing, and Radio layers | MAC layer + Packet Timing only | Only MAC layer features |
| Machine Learning Model | LSTM + DTW + Random Forest for time-series classification | Random Forest classifier | Support Vector Machine (SVM) |
| Time-Series Analysis | Yes, uses DTW and LSTM to track device behavior over time | No time-series tracking | No time-series tracking |
| Adversarial Attack Resistance | High resistance to MAC Spoofing, Timing Manipulation, and Feature Injection | Low resistance, vulnerable to MAC Spoofing | Moderate resistance, partially resistant to MAC Spoofing |
| Scalability | High, tested on 10+ devices, adaptable to large-scale deployments | Medium, tested on 10 devices | Low, tested on 5 devices |
| Classification Accuracy | 97.0% under normal conditions, 95.0% under adversarial attacks | 92.5% under normal conditions | 94.2% under normal conditions |

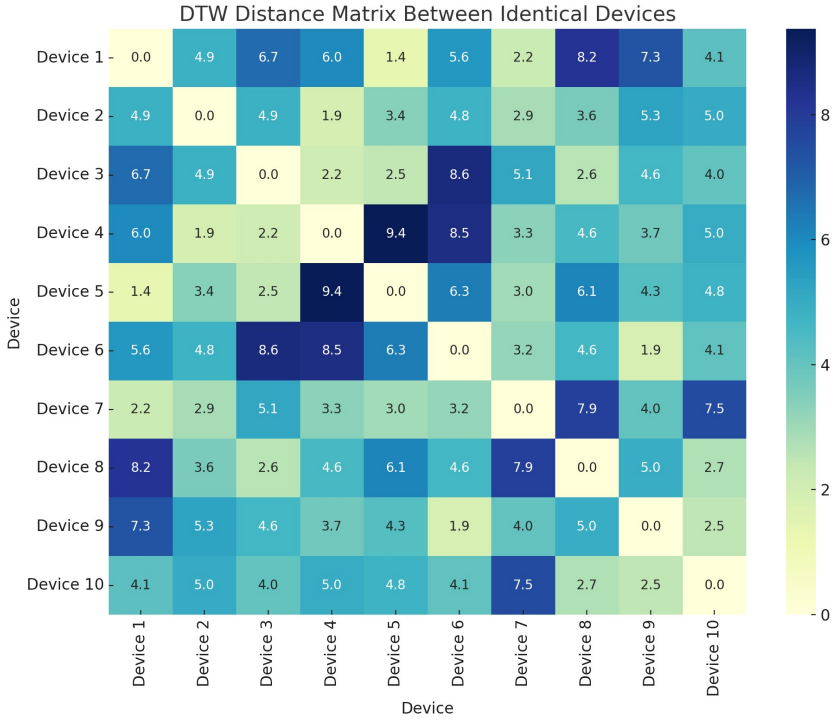


*Fig. 4. DTW Distance Matrix between Different Devices*

It can be observed that some devices exhibit similar values in key features such as RTT, TCP Sequence Numbers, or RSSI, leading to confusion between devices with similar network behavior. Similarly, variations in the RSSI or SNR values due to environmental factors, such as interference, also lead to occasional misclassifications, especially between devices physically close to each other. However, the overall performance in terms of a high number of correct predictions suggests that the combination of features used in the experiment provides strong differentiation between identical devices.

The above results only showcase the performance of the proposed model in a non-attack scenario. The following experiments highlight the performance of the proposed device identification and fingerprinting model under adversarial attack scenarios.

The same network setup was considered. All devices were configured with identical firmware and equipped with sensors (e.g., temperature, humidity, and light), reporting data every 60 seconds over a Wi-Fi network.

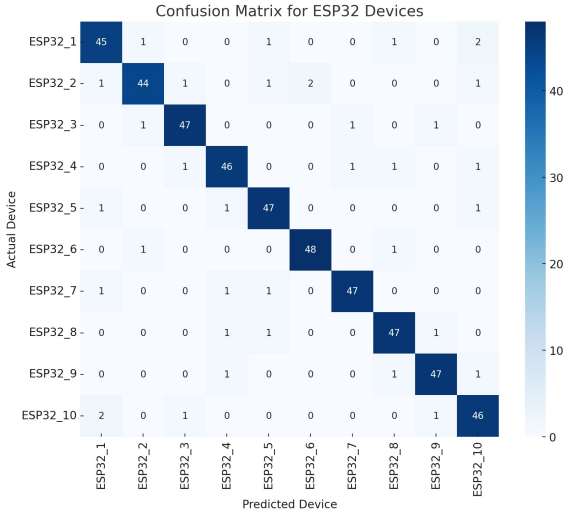


*Fig. 5. Confusion Matrix for ESP32 Devices*

The fingerprinting model uses multi-layer feature extraction (MAC layer, radio characteristics, transport layer, and timing features), Dynamic Time Warping (DTW) for time-series alignment, and LSTM-based temporal modeling to generate unique device fingerprints. The evaluation objective was to determine how well the model performs under adversarial attacks designed to disrupt feature integrity or mimic legitimate device behavior. The model's classification accuracy was measured across five experimental runs for each attack scenario.

In a MAC spoofing attack, as shown in

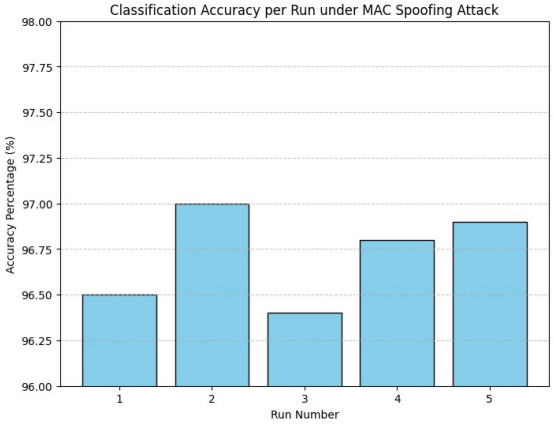


Figure 6, an adversary changes the MAC address of their device to match the MAC address of a legitimate device. Since the MAC address is often treated as a unique identifier in network protocols, spoofing allows the attacker to impersonate the legitimate device.

*Fig. 6. Classification Accuracy under MAC Spoofing Attack*

The attacker identifies the MAC address of the legitimate device (e.g., through passive sniffing or probing). They configure their device to use the same MAC address. The attacker attempts to interact with the network, appearing as a legitimate device. This attack can confuse models that rely heavily on the MAC layer for fingerprinting, leading to misclassifications. The proposed fingerprinting model maintains high accuracy (95.6%-96.0%) even under MAC spoofing attacks, owing to its multi-layer feature extraction and advanced processing techniques. The model does not rely solely on MAC addresses. Instead, it incorporates features from radio (RSSI, SNR), transport (RTT, TCP sequence gaps), and timing (IAT, clock skew) layers. While attackers can spoof a MAC address, they cannot alter the device's underlying hardware characteristics, such as signal strength variations or clock skew. DTW aligns time-series data, such as RTT or IAT, between devices. If the attacker attempts to mimic these features, subtle inconsistencies in timing patterns will be detected.

DTW ensures that even devices with similar feature distributions can be distinguished by their temporal behavior. The LSTM network

captures long-term feature dependencies, such as variations in probe request intervals or inter-packet delays.

Attackers cannot easily replicate the nuanced temporal patterns legitimate devices generate over extended periods. Fusing features from multiple layers creates redundancy, meaning that even if one feature (e.g., MAC address) is spoofed, other features (e.g., RTT, clock skew) remain unique. This makes it extremely difficult for attackers to impersonate a legitimate device fully. The MAC spoofing attack results in only a 1.2% drop in accuracy compared to no attack, demonstrating the model's resilience. Using a multi-layer approach, the model ensures that attacks on a single feature (e.g., MAC spoofing) cannot significantly degrade performance. Temporal modeling and time-series alignment add robustness by capturing behavioral patterns that are hard to mimic.

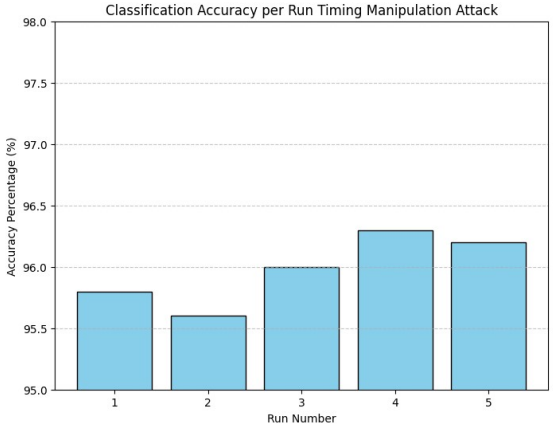


*Fig.7.Classification Accuracy under Timing Manipulation Attack*

The timing manipulation attack targets the time-dependent features used in the fingerprinting model, such as Round-Trip Time (RTT), Inter-Packet Arrival Time (IAT), Probe Request Interval, and Clock Skew Variations, as shown in Fig. 7. In this attack, an adversary deliberately modifies these timing parameters to mimic another device's behavior, attempting to confuse the fingerprinting model and cause misclassification. The attacker introduces artificial delays to alter RTT, manipulates packet transmission intervals to adjust IAT, or adjusts probe request intervals to match another device's pattern. However, despite these manipulations, the model maintains an accuracy of 95.6% to 96.2%, only experiencing a marginal drop of 1.3% from the baseline accuracy. This minimal performance degradation is due to the multi-layer feature extraction and advanced processing techniques used in the model. DTW ensures that long-term time-series inconsistencies are detected even if short-term timing behavior is mimicked.

LSTM-based temporal modeling captures device-specific timing behaviors, preventing complete impersonation. Additionally, clock skew variations, which are hardware-dependent and difficult to manipulate, provide further resilience. Lastly, cross-feature validation ensures that any inconsistencies in manipulated features are flagged, maintaining the integrity of device identification. These mechanisms collectively make the fingerprinting model highly robust against timing-based adversarial attacks while ensuring accurate device differentiation.

The feature injection attack targets the core distinguishing features used in device fingerprinting. Attackers mislead the classifier by introducing manipulated values into RSSI, SNR, TCP Sequence Gaps, and Transport Layer Parameters. The primary objective of this attack is to create overlapping feature distributions across multiple devices, making them appear similar and causing misclassification. Attackers achieve this by injecting false RSSI readings, altering signal strength patterns, or modifying TCP sequence behavior. Despite these adversarial modifications, the proposed fingerprinting model maintains an accuracy of 94.3% to 95.2%, experiencing only a marginal 2.0% performance drop from the baseline accuracy. This is due to the feature validation and redundancy mechanisms embedded in the model. Correlation-based feature validation ensures that injected values do not disrupt classification by detecting inconsistencies in feature distributions. LSTM-based temporal modeling captures device-specific feature trends over time, making it difficult for injected data to replicate legitimate behavior fully. Dynamic Time Warping (DTW) also aligns time-series features, ensuring that fabricated feature values do not match real devices' patterns. Lastly, anomaly detection mechanisms flag statistical deviations, ensuring that injected feature values are detected before corrupting the classification process. These robust defenses prevent significant accuracy degradation, keeping device fingerprinting reliable under feature injection attacks.

Lastly, the performance of the proposed approach is compared with two other existing works that address the IoT device fingerprinting problem. The comparison includes multiple parameters such as features considered, machine learning models used, adversarial attack resistance, scalability, and accuracy. The table 4 shows the

comparison where the LSTM-based time-series modeling significantly improves attack resilience. Multi-layer feature extraction is essential for differentiating identical IoT devices under adversarial settings. The proposed approach maintains high accuracy even under attack conditions, proving its efficiency.

## 5. CONCLUSION AND FUTURE SCOPE

In this paper, we introduce a novel approach, DIFT-VAR, for fingerprinting and differentiating between identical devices by leveraging multi-layer feature extraction, Dynamic Time Warping (DTW), and LSTM-based temporal modeling. The method relies on extracting features such as RSSI, SNR, TCP sequence numbers, RTT, and probe request intervals to capture the unique behavior of each device. DTW is used to align the time-series data, addressing small variations in timing due to environmental factors. On the other hand, LSTM captures long-range temporal dependencies, providing unique sequences for each device. The fusion of these features forms a robust feature vector used to train a Random Forest classifier, achieving consistently high accuracy (97%) across multiple runs. The confusion matrix shows minimal misclassifications, confirming the model's reliability in distinguishing between devices. This approach demonstrates that even identical devices, when analyzed based on subtle network behavior and timing differences, can be effectively fingerprinted. The work opens up possibilities for enhanced device authentication, anomaly detection, and security in IoT networks, providing a scalable and robust solution for real-world IoT device identification challenges, particularly where devices have identical hardware and software configurations. Models would do several steps to satisfy the authentication requirements of devices within a network and, in doing so, could possibly have overhead. There could be some restrictions present, but it is a usual and necessary procedure to maintain an adequate level of security. The proposed model operates on very similar devices, which, when operational, only have minute differences. The model has the ability to produce false positives since it relies upon these minute differences to identify them. This is acknowledged as a one of the limitations of the proposed. However, through the collection of more real-time traffic and taking a closer examination of these minute differences, this can be minimized, and false positive rates may be limited.

Future work could explore the use of additional features such as power consumption and energy patterns, as well as extending this approach to larger-scale IoT deployments.

## REFERENCES:

[1] Q. Xu, R. Zheng, W. Saad, and Z. Han, “Device fingerprinting in wireless networks: Challenges and opportunities,” IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 94–104, 2016.

[2] V. Thangavelu, D. M. Divakaran, R. Sairam, S. S. Bhunia, and M. Gurusamy, “Deft: A distributed iot fingerprinting technique,” IEEE Internet of Things Journal, vol. 6, no. 1, pp. 940–952, 2019.

[3] A. Chaman, J. Wang, J. Sun, H. Hassanieh, and R. Roy Choudhury,“Ghostbuster: Detecting the presence of hidden eavesdroppers,” in Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, ser. MobiCom ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 337–351. [Online]. Available: https://doi.org/10.1145/3241539.3241580

[4] C. Shen and J. Huang, “EarFisher: Detecting wireless eavesdroppers by stimulating and sensing memory EMR,” in 18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21). USENIX Association, Apr. 2021, pp. 873–886. [Online]. Available: https://www.usenix.org/conference/nsdi21/presentation/shen.

[5] T. Liu, Z. Liu, J. Huang, R. Tan, and Z. Tan, “Detecting wireless spy cameras via stimulating and probing,” in Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, ser. MobiSys ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 243–255. [Online]. Available: https://doi.org/10.1145/3210240.3210332.

[6] E. Rodr´ıguez, P. Valls, B. Otero, J. J. Costa, J. Verd´u, M. A. Pajuelo,and R. Canal, “Transfer-learning-based intrusion detection framework in iot networks,” Sensors, vol. 22, no. 15, 2022. [Online].Available: https://www.mdpi.com/1424-8220/22/15/5621.

[7] A. Nazerian and F. Parastar, “Passive iot device fingerprinting using wifi,” in Proceedings of the 12th ACM Wireless of the Students, by the

students, and for the students (S3) Workshop, er. S3 '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 6–7. [Online]. Available: https://doi.org/10.1145/3477087.3478381.

[8] T. Kohno, A. Broido, and K. Claffy, "Remote physical device finger-printing," in 2005 IEEE Symposium on Security and Privacy (SP'05),2005, pp. 211–225.

[9] J. Kurmi and R. Matam, "Device identification in iot networks using network trace fingerprinting," in 2022 International Conference on Smart Applications, Communications and Networking (SmartNets), 2022, pp. 1–6.

[10] V. Visoottiviseth, P. Sakarin, J. Thongwilai, and T. Choobanjong, "Signature-based and behavior-based attack detection with machine learning for home iot devices," in 2020 IEEE REGION 10 CONFERENCE (TENCON),2020, pp. 829–834.

[11] N. Yousefnezhad, A. Malhi, and K. Framling, "Security in product lifecycle of iot devices: A survey," Journal of Network and Computer Applications, vol. 171, p. 102779, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804520302538.

[12] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa,N. O. Tippenhauer, and Y.Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in Proceedings of the Symposium on Applied Computing,ser. SAC'17. New York, NY, USA: Association for Computing Machinery, 2017, p. 506–509. [Online]. Available: https://doi.org/10.1145/3019612.3019878.

[13] M. Miettinen, S. Marchal, I. Hafeez, N.Asokan, A.-R. Sadeghi,and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), 2017, pp.2177–2184.

[14] A. Sivanathan, D. Sherratt, H. H. Gharakheili A.Radford, C. Wi-jenayake, A. Vishwanath, and V. Sivaraman, "Characterizing and classifying iot traffic in smart cities and campuses," in 2017 IEEE Conference on Computer Communications Workshops (INFOCOMWKSHPS), 2017, pp. 559–564.

[15] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic," in ICC 2019 - 2019 IEEE International Conference on Communications (ICC), 2019, pp. 1–7.

[16] M. Mainuddin, Z. Duan, Y. Dong, S. Salman, and T. Taami, "Iot device identification based on network traffic characteristics," in GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 2022, pp.6067–6072.

[17] L. Xie, L. Peng, J. Zhang, and A. Hu, "Radio frequency fingerprint identification for internet of things: A survey," Security and Safety, vol.3, p. 2023022, 2024.

[18] A. Jagannath, J. Jagannath, and P. S. P. V. Kumar, "A comprehensive survey on radio frequency (rf) fingerprinting: Traditional approaches, deep learning, and open challenges," Computer Networks, vol. 219, p. 109455, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622004893.

[19] M. K¨ose, S. Tas¸cio˘glu, and Z. Telatar, "Rf fingerprinting of iot devices based on transient energy spectrum," IEEE Access, vol. 7, pp. 18 715–18 726, 2019.

[20] G. Baldini, G. Steri, R. Giuliani, and C. Gentile, "Imaging time series for internet of things radio frequency fingerprinting," in 2017 International Carnahan Conference on Security Technology (ICCST), 2017, pp. 1–6

[21] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device finger-printing," IEEE Transactions on Dependable and Secure Computing, Vol. 2, no. 2, pp. 93–108, 2005.

[22] C. Shang, J. Cao, T. Zhu, Y. Zhang, B. Niu, and H. Li, "Cadfa: A clock skew-based active device fingerprint authentication scheme for class-1 iot devices," IEEE Systems Journal, 2024.

[23] H. Wang, D. Eklund, A. Oprea, and S. Raza, "Fl4iot: Iot device fingerprinting and identification using federated learning," ACM Trans. Internet Things, vol. 4, no. 3, Jul. 2023. [Online]. Available: https://doi.org/10.1145/3603257.

[24] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," arXiv preprint arXiv:1408.1416, 2014.

[25] F. Z. Fagroud, H. Toumi, E. H. B. Lahmar, K. Achtaich, S. E. Filali, and Y. Baddi, "Connected devices classification using feature selection

with machine learning," IAENG International Journal of Computer Science, vol. 49, no. 2, 2022.

[26] M. Noorafiza, K. Ishak, H. Maeda, M. Shiratori, T. Kinoshita, and R. Uda, "Characteristic patterns of timestamps from android operating system on mobile device and virtual machine." IAENG International Journal of Computer Science, vol. 43, no. 2, 2016.

[27] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of iot devices in the cyberspace," Computer Networks, vol. 148, pp. 318–327,2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128618306856.

[28] Qian Lu , Zaikai Yang , Hanlin Zhang "MRFE: A Deep-Learning-Based Multidimensional Radio Frequency Fingerprinting Enhancement Approach for IoT Device Identification IEEE INTERNET OF THINGS JOURNAL, VOL. 11, NO. 18, 15 SEPTEMBER 2024 [Online] Available:https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10556754