

Article

Wireless Signal Fingerprinting Framework Based on Emphasized Spectral Features for IoT Device Authentication

Hyeon Park , Geumhwan Cho *  and TaeGuen Kim * 

Department of Cybersecurity, Korea University, 2511 Sejong-ro, Sejong 30019, Republic of Korea;
rainbow00000@korea.ac.kr

* Correspondence: geumhwan@korea.ac.kr (G.C.); tg_kim@korea.ac.kr (T.K.)

Abstract

Bluetooth Low Energy (BLE) is widely used in Internet of Things (IoT) devices due to its low power consumption and efficient wireless communication. However, BLE-based systems remain vulnerable to signal-level attacks, such as spoofing and signal forgery, which allow adversaries to impersonate legitimate devices and compromise system security. Existing security approaches mainly rely on cryptographic mechanisms or protocol-level features, while conventional signal fingerprinting methods often fail to capture subtle device-specific variations across the frequency spectrum. We propose a deep-learning-based BLE signal fingerprinting framework that uses emphasized spectral data to enhance device authentication. The proposed framework selectively highlights frequency regions exhibiting pronounced hardware-dependent variations using a hybrid filter bank design and extracts spectral features for anomaly-based device identification. Experimental evaluations conducted on BLE signals collected from multiple devices demonstrate that the proposed approach outperforms conventional methods, achieving superior authentication performance. By leveraging emphasized frequency-domain characteristics, we provide an effective authentication method for BLE-based IoT environments.

Keywords: Bluetooth Low Energy; BLE fingerprinting; RF fingerprinting; physical-layer authentication; hybrid filter bank; deep-learning-based authentication

MSC: 68M25; 68T07; 94A12; 94A13; 62H30; 94A62



Academic Editor: Jonathan Blackledge

Received: 11 May 2026

Revised: 18 June 2026

Accepted: 22 June 2026

Published: 1 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

The rapid expansion of the Internet of Things (IoT) has led to widespread adoption of Bluetooth Low Energy (BLE) as a core wireless communication protocol in modern IoT environments. BLE is particularly attractive for resource-constrained devices due to its low power consumption and efficient data transmission, enabling its deployment across diverse application domains such as smart homes, healthcare monitoring, logistics, and retail systems [1]. As BLE-based IoT devices increasingly handle sensitive personal data and support safety critical services, ensuring reliable and robust device authentication has become an essential security requirement.

Despite security mechanisms in BLE, including encryption and pairing protocols, practical deployments remain vulnerable to various attacks that exploit the wireless nature of BLE communication. In particular, attackers can impersonate legitimate devices by relaying, spoofing, or manipulating wireless signals without modifying higher-layer protocol data. For example, in [2], researchers experimentally demonstrated BLE replay

attacks using software-defined radio platforms, where captured advertising messages and pairing sequences were retransmitted to generate false pairing notifications and fabricated device information without modifying higher-layer protocol data or compromising cryptographic credentials. These real-world attack scenarios highlight fundamental limitations of conventional BLE security mechanisms that primarily rely on protocol-level authentication because radio-frequency-based replay and spoofing attacks can exploit the wireless nature of BLE communications.

Conventional BLE security mechanisms primarily rely on cryptographic pairing and key-establishment procedures, link-layer encryption, address randomization, and protocol-level access control. These mechanisms are essential for protecting data confidentiality and limiting unauthorized access, but they do not directly verify whether the received radio signal originates from the genuine physical transmitter. Therefore, physical-layer authentication based on radio-frequency (RF) fingerprints can serve as a complementary defense by examining hardware-dependent signal characteristics that are difficult to reproduce through protocol-level manipulation alone.

Existing countermeasures against such threats have focused on cryptographic enhancements, protocol-level defenses, or traffic and timing-based anomaly detection. While these approaches offer partial protection, they often fail to address signal impersonation attacks that preserve protocol correctness while altering physical-layer characteristics. Signal fingerprinting has therefore emerged as a promising complementary approach, as it leverages device-specific imperfections introduced by hardware components during signal transmission. However, many existing fingerprinting methods analyze fixed spectral features or uniformly processed frequency components, which limits their ability to capture subtle and localized variations that distinguish devices under realistic conditions.

Our proposed BLE signal fingerprinting framework addresses these limitations by focusing on frequency domain characteristics that exhibit pronounced device-specific variations. BLE signals transmitted by different devices exhibit subtle but consistent differences in their frequency spectra due to hardware-dependent factors such as oscillators, amplifiers, and modulation circuits. These differences are not uniformly distributed across the spectrum but are concentrated in specific frequency regions, while other regions exhibit relatively similar patterns across devices. To exploit this observation, we introduce a frequency-domain highlighting strategy to selectively emphasize informative regions while suppressing less discriminative components. In frequency domain signal analysis, filter banks are commonly employed to partition the spectrum and extract representative spectral features for subsequent processing. Conventional filter banks are typically designed to provide uniform or perceptually motivated frequency resolution and are applied identically across all frequency regions. In contrast, our proposed approach defines a device-oriented filter bank structure in advance, where frequency regions exhibiting higher inter-device variability are assigned to denser filtering resolution. This emphasized filter bank is constructed by combining an inverse mel scale filter bank for highlighted regions with a linear scale filter bank for non-highlighted regions, enabling fine grained analysis of discriminative spectral components while preserving overall spectral structure. By applying this filter bank to BLE signals, emphasized spectral data are extracted and used as input features for device authentication.

Once the emphasized spectral representation is obtained, authentication can be performed by analyzing how closely incoming signals match the learned characteristics of legitimate devices. The extracted spectral data are fed into a learning-based anomaly detection model, allowing legitimate device signals to be distinguished from forged or manipulated signals based on reconstruction behavior. This design enables authentication to be performed directly at the physical layer without modifying existing BLE protocols or

requiring additional user interaction. To evaluate the effectiveness of the proposed framework, extensive experiments were conducted using BLE signals collected from multiple transmission devices. Several machine learning models and filter bank configurations were evaluated to identify the most effective combination for BLE signal fingerprinting. The experimental results show that an autoencoder-based deep neural network, when trained on emphasized spectral data extracted using the inverse mel scale filter bank, achieves superior authentication performance compared to conventional spectrum-based and uniformly filtered approaches. These results demonstrate that selectively emphasizing frequency domain characteristics significantly enhances the discriminability of BLE signal fingerprints.

The main contributions of our work can be summarized as follows:

- A deep learning-based BLE signal fingerprinting framework is proposed, achieving an F1-score of 0.9927 for device authentication by leveraging physical layer signal characteristics.
- A frequency-domain highlighting strategy is introduced to emphasize discriminative frequency regions where device-specific variations are more pronounced, thereby improving the separability of BLE signals transmitted by different devices.
- A hybrid filter bank design is employed, combining an inverse mel scale filter bank for highlighted regions and conventional filter banks for non-highlighted regions, enabling effective extraction of informative spectral features.
- Extensive experimental evaluations are conducted using BLE signals collected from 40 devices with multiple filter bank configurations and five learning models, demonstrating that an autoencoder-based deep neural network (DNN) trained on highlighted spectral data achieves superior performance in BLE signal identification.

The remainder of our work is organized as follows. Section 2 provides background information on BLE, its applications, and associated security threats. Section 3 reviews related work on BLE security and device fingerprinting. Section 4 describes the proposed BLE signal fingerprinting framework in detail, including feature extraction, frequency highlighting, and model architecture. Section 5 presents the experimental setup, evaluation methodology, and performance analysis. Finally, Section 6 concludes the paper and discusses potential directions for future research.

2. Background

This section provides an overview of the security mechanisms and privacy protection features implemented in Bluetooth Low Energy, as well as security threats that arise from its design and operational characteristics.

Bluetooth Low Energy is a wireless communication protocol designed for low power consumption and efficient short range data transmission in the 2.4 GHz industrial, scientific, and medical (ISM) band. Its lightweight communication model and rapid connection establishment make BLE suitable for battery powered devices requiring reliable and timely data exchange. As a result, BLE has been widely adopted in applications such as smart homes, wearable devices, health monitoring systems, and industrial automation, where energy efficiency and stable connectivity are essential.

As BLE-based systems are deployed across increasingly diverse application domains, security and privacy concerns have become critical considerations. BLE incorporates several security mechanisms, including encryption and device pairing procedures, to protect data confidentiality and control access between devices. These mechanisms are intended to provide baseline protection against unauthorized communication while maintaining usability in resource-constrained environments.

Despite these protections, the wireless nature of BLE communication exposes systems to a range of security threats that cannot be fully mitigated by conventional protocol-level

defenses. Common attack vectors include passive eavesdropping, Man-in-the-Middle (MitM) attacks during the pairing process, replay-based retransmission attacks, and unauthorized pairing attempts. Passive eavesdropping occurs when an attacker intercepts wireless transmissions before encryption is applied or during insecure pairing phases. MitM attacks exploit weaknesses in pairing procedures, allowing an attacker to intercept and manipulate communications while appearing legitimate to both endpoints. Retransmission attacks involve capturing valid messages and replaying them at a later stage, potentially triggering unintended actions. Unauthorized pairing attacks exploit usability-focused pairing methods to gain access without proper authentication.

Several studies have demonstrated that such attacks are feasible in real-world BLE deployments. For example, experimental analyses in [3] demonstrated that pairing downgrade attacks allow adversaries to force BLE devices into less secure pairing modes, thereby weakening authentication guarantees and enabling subsequent exploitation. These attack scenarios highlight that, even when protocol-level security mechanisms are in place, BLE communications remain vulnerable to attacks that exploit the wireless characteristics of signal transmission.

A key limitation of conventional defenses is that they primarily rely on cryptographic verification and protocol correctness, which are insufficient against attacks that preserve protocol behavior while altering physical layer characteristics. As a result, attackers can exploit hardware-independent assumptions to mimic legitimate devices at the signal level, bypassing existing protections without breaking encryption or authentication protocols.

To address this gap, physical layer-based authentication approaches have gained attention as complementary security mechanisms. By analyzing intrinsic signal characteristics introduced by hardware imperfections, signal fingerprinting enables device identification beyond protocol level information. Such approaches are particularly effective against spoofing and replay-based attacks, as forged signals often fail to replicate subtle device-specific frequency characteristics.

The authentication approach adopted in this work targets these signal-level threats by operating transparently alongside existing BLE mechanisms. In particular, it is designed to enhance the security of usability-focused pairing methods such as Just Works, which lack explicit user authentication but are widely deployed in practical systems. By enabling authentication based on signal fingerprinting without requiring user interaction or protocol modification, the proposed approach strengthens BLE security while preserving deployment flexibility and user convenience.

3. Related Work

This section reviews prior studies on BLE security from multiple perspectives, including protocol-level vulnerabilities, MitM attack detection, spoofing and replay attack mitigation, and device fingerprinting. The reviewed works are grouped by research focus to clarify their respective approaches and limitations, and to position the proposed method within the broader landscape of BLE security research.

3.1. Protocol-Level Vulnerabilities and Privacy Weaknesses

Several studies have investigated inherent vulnerabilities in the BLE specification and its real-world implementations, particularly from privacy and traceability perspectives. Wu et al. [4] showed that device and user tracking remains feasible despite privacy-preserving mechanisms such as address randomization. Perri et al. [5] and Locatelli et al. [6] further demonstrated that information exposed during the BLE device discovery process alone can be exploited to identify and trace devices. Zuo et al. [7] revealed that static universally unique identifiers (UUIDs) embedded in mobile applications enable

automatic fingerprinting of vulnerable BLE IoT devices. These works collectively highlight fundamental privacy and security weaknesses rooted in BLE protocol design. However, they primarily rely on protocol metadata or higher-layer information and do not exploit fine-grained physical-layer signal characteristics for device authentication.

3.2. MitM Attack Detection in BLE Environments

Another research direction focuses on detecting MitM attacks in BLE environments by analyzing changes in communication behavior. Yaseen et al. [8] highlighted that simplified BLE pairing modes facilitate MitM attacks and proposed an anomaly-based detection framework targeting eHealthcare BLE systems. Yurdagul and Sencar [9] leveraged response-time variations introduced by MitM attacks to design a lightweight detection mechanism, while Lahmadi et al. [10] employed machine-learning-based traffic reconstruction and classification to identify MitM behavior. Although these approaches demonstrate that temporal and traffic-pattern changes can indicate MitM attacks, they are often sensitive to network conditions and adversarial evasion.

3.3. Physical-Layer-Based Security for Spoofing and Replay Attack Detection

Physical-layer-based security approaches, particularly RF fingerprinting, have also been explored to detect spoofing and replay attacks. Galtier et al. [11] proposed a power spectral density (PSD)-based RF fingerprinting technique for IoT spoofing detection. Wu et al. [12] introduced BlueShield, which detects spoofing attacks by analyzing BLE protocol behavior and traffic patterns. Abu Al-Haija and Alsulami [13] applied deep learning to RF signals to detect replay attacks in remote keyless vehicle systems. While these studies demonstrate the effectiveness of physical-layer signal characteristics for attack detection, most rely on limited spectral representations or predefined features.

3.4. BLE Device Fingerprinting for Identification and Authentication

Extensive research has further examined BLE device fingerprinting for identification and authentication. Bonavolontà et al. [14] improved BLE fingerprinting techniques for secure deployment in passive entry passive start (PEPS) vehicle systems, and Liu et al. [15] proposed bidirectional device identification based on RF fingerprint reciprocity. Zhang et al. [16] presented a comprehensive study of physical-layer fingerprinting for wireless security, while Stoian et al. [17] enhanced BLE fingerprinting performance by incorporating instantaneous frequency features. Zhang et al. [18] showed that BLE devices can be identified using link-layer broadcast packet characteristics alone, and Sun and Dang [19] proposed FingerBLE, a BLE-specific fingerprinting scheme. More recently, Shen et al. [20] introduced a federated RF fingerprint identification framework based on unsupervised contrastive learning. Despite these advances, most prior studies adopt fixed feature extraction schemes or limited spectral representations.

3.5. Deep-Learning-Based RF Fingerprinting Studies

Recent RF fingerprinting and specific emitter identification (SEI) studies have increasingly focused on robust deep-learning frameworks under practical deployment conditions. For example, channel-robust SEI has been investigated through single-source domain generalization and shortcut-learning mitigation, showing that deep models may rely on channel-dependent shortcuts rather than stable hardware-specific fingerprints [21]. Open-world RF fingerprint identification (RFFI) has also been studied to identify known emitters while detecting unknown or novel emitters, and transformer-based architectures have been introduced to improve RF feature extraction [22]. In addition, cross-domain RFFI methods have explored adaptive semantic augmentation and multi-resolution spectrogram decom-

position to improve robustness against distribution shifts caused by different acquisition conditions [23].

These recent studies indicate that robust feature representation remains a central issue in RF-based device authentication. Unlike these general SEI/RFFI frameworks, this study focuses on BLE device authentication using emphasized spectral representations and hybrid filter-bank features. The proposed approach is designed to capture device-discriminative spectral regions in BLE signals and evaluate their effectiveness in an authentication-oriented normal/abnormal setting.

3.6. System-Level BLE and IoT Security Approaches

Finally, broader system-level perspectives on BLE and IoT security have been explored. Pallavi and Narayanan [24] surveyed practical attacks against BLE-based IoT devices and discussed their security implications. Lacava et al. [25] proposed an intrusion detection system for Bluetooth Mesh networks using real traffic and experimental evaluation, and Gu et al. [26] introduced IoTGaze, a framework that enforces IoT security policies through wireless context analysis. While these studies provide valuable insights into system- and network-level defenses, they do not address fine-grained device authentication based on physical-layer signal fingerprints.

3.7. Positioning of the Proposed Method

In contrast to existing approaches, our work focuses on device-level authentication by exploiting frequency-domain physical-layer fingerprints. By dynamically identifying and emphasizing discriminative frequency regions through a filter-bank-based design, the proposed method enables more robust and fine-grained BLE device fingerprinting that complements prior protocol-, traffic-, and system-level security studies.

4. Proposed Deep-Learning-Based Framework

This section introduces our BLE device authentication framework, which is designed around emphasized-spectral-data-based signal fingerprinting. The framework is motivated by the observation that BLE transmission devices exhibit device-dependent spectral characteristics that are not uniformly distributed across the frequency spectrum. Rather than processing all frequency components equally, the framework focuses on selectively emphasizing frequency regions that contain distinctive device-specific information, enabling robust authentication based on physical-layer signal properties.

At a high level, the proposed framework consists of five stages: BLE signal acquisition, signal preprocessing, emphasized spectral feature extraction, autoencoder-based representation learning, and anomaly-score-based authentication. The preprocessing stage applies pre-emphasis, framing, windowing, and fast Fourier transform (FFT) to convert raw BLE waveforms into frequency-domain representations. The feature extraction stage then constructs emphasized filter-bank features by assigning higher spectral resolution to frequency regions with strong device-dependent variations. Finally, the autoencoder-based DNN (AE-DNN) and autoencoder-based one-dimensional convolutional neural network (AE-1D CNN) models learn normal-device feature distributions, and reconstruction errors are converted into authentication scores. Detailed network parameters and thresholding procedures are described in Sections 4.2.1 and 4.2.3.

The framework is organized around a common feature extraction pipeline that is shared across phases, while threshold determination and authentication are performed in separate training and testing phases, respectively. Its operation is divided into a training phase and a testing phase, each of which performs a specific role while sharing a common signal processing pipeline.

During the training phase, the framework takes as input a collection of normal BLE signal data obtained from a given device. Each input signal is first processed through a sequence of signal preprocessing steps, including pre-emphasis, framing, windowing, and frequency-domain transformation, to obtain a spectral representation. Based on this representation, frequency regions that exhibit pronounced deviations relative to other regions are identified. These regions are designated as emphasis sections, while the remaining regions are treated as non-emphasized sections.

To extract informative spectral features, the framework constructs a device-specific filter bank based on the identified emphasis sections. An inverse Mel scale filter bank is applied to the emphasis sections to enhance spectral resolution in regions with strong device-dependent variations, whereas a linear scale filter bank is applied to the non-emphasized sections to preserve global spectral characteristics. By applying this combined filter bank to the frequency-domain signal representation, the framework produces emphasized spectral data in the form of compact spectral feature vectors.

These spectral feature vectors serve as input to a deep learning model based on an autoencoder architecture. The autoencoder is trained using normal data to learn intrinsic reconstruction patterns associated with legitimate device signals. In parallel, a separate dataset is used to analyze reconstruction behavior and determine a decision threshold. Reconstruction errors are computed by measuring the mean squared error between the input spectral features and their reconstructed outputs, and an optimal threshold is selected based on reconstruction error statistics to distinguish normal behavior from abnormal behavior.

In the testing phase, incoming BLE signals are processed using the same preprocessing and feature extraction pipeline established during the training phase, with the exception that the configuration of emphasized frequency sections is not recomputed and the predefined emphasis settings are directly applied. The signal is transformed into the frequency domain, and spectral features are extracted using the filter bank determined in the training phase, without redefining emphasis regions or filter parameters. The resulting spectral feature vectors are then fed into the trained autoencoder to compute reconstruction errors. Authentication is performed by comparing these reconstruction errors against the predefined threshold, allowing the framework to determine whether the observed signal behavior is consistent with legitimate device characteristics or indicative of anomalous activity.

Through this design, the framework establishes a clear flow of data from raw BLE signals to authentication decisions, while maintaining consistency between the training and testing phases. By explicitly linking emphasized frequency-domain feature extraction with learning-based reconstruction analysis, the framework enables effective BLE device authentication based on signal fingerprinting. The overall workflow of the framework, including the interaction between its processing stages and the flow of input and output data across phases, is illustrated in Figure 1.

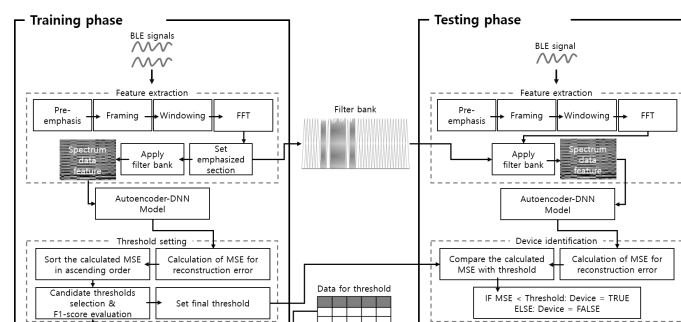


Figure 1. Overall flow of the proposed device authentication framework.

4.1. Feature Extraction

The feature extraction process is explicitly designed to exploit emphasized frequency characteristics. The feature extraction module consists of several sequential steps, including pre-emphasis, framing, windowing, FFT, application of an emphasized filter bank, and spectral data extraction. The input to this process is the raw time-domain BLE signal acquired from a transmission device, while the output is a compact spectral feature vector that captures device-specific frequency characteristics.

Due to subtle hardware-dependent variations, BLE transmission devices produce signals with distinct frequency responses. Frequency-domain analysis of signals collected from multiple BLE devices reveals that these differences are not uniformly distributed across the spectrum but are concentrated in specific frequency intervals. These intervals are defined as emphasis sections, whereas the remaining frequency ranges are classified as general sections. By explicitly separating these two regions, the proposed framework enhances discriminative frequency components while preserving global spectral information.

Based on this separation, an emphasized filter bank is applied to the emphasis section to extract spectral features with high inter-device variability. Simultaneously, a linear scale filter bank is applied to the general section to retain overall spectral structure. The outputs of the two filter banks are subsequently merged, producing a unified spectral feature representation that serves as the final output of the feature extraction module.

The overall processing flow of feature extraction is summarized in Algorithm 1. Starting from BLE signal acquisition, the signal undergoes pre-emphasis, framing, and windowing, followed by FFT to obtain a frequency-domain representation. Next, frequency intervals exhibiting distinctive characteristics across devices are identified, and the corresponding inverse Mel scale and linear scale filter banks are constructed for the emphasis and general sections, respectively. These filter banks are then jointly applied to the FFT-transformed signal to generate the emphasized spectral features used in subsequent analysis. Each step for feature extraction is explained in the following subsections.

Algorithm 1 Feature Extraction

```

1: Initialize devices  $\mathcal{D}$ 
2:  $S \leftarrow \{\}$ 
3:  $R \leftarrow \{\}$ 
4:  $B \leftarrow \{\}$ 
5:  $F \leftarrow \{\}$ 
6: for each  $d \in \mathcal{D}$  do
7:    $S[d] \leftarrow \text{Collect\_BLE\_signals}(d)$ 
8: end for
9: for each  $d \in \mathcal{D}$  do
10:   $s \leftarrow S[d]$ 
11:   $s \leftarrow \text{PreEmphasis}(s)$ 
12:   $f \leftarrow \text{Framing}(s)$ 
13:   $w \leftarrow \text{Windowing}(f)$ 
14:   $X \leftarrow \text{FFT}(w)$ 
15:   $\Phi \leftarrow \text{AnalyzeFrequencies}(X)$ 
16:   $R[d] \leftarrow \text{FindEmphasizedRegions}(\Phi, W)$ 
17:   $r \leftarrow R[d]$ 
18:   $F_e \leftarrow \text{DesignInverseMelFilters}(r)$ 
19:   $r_n \leftarrow \text{GetNonEmphasizedRegions}(\Phi, r)$ 
20:   $F_l \leftarrow \text{DesignLinearFilters}(r_n)$ 
21:   $b \leftarrow \text{MergeFilters}(F_e, F_l)$ 
22:   $B[d] \leftarrow b$ 
23:   $c \leftarrow \text{ApplyFilterBank}(b, X)$ 
24:   $F[d] \leftarrow c$ 
25: end for
26: return  $B, F$ 

```

$\triangleright S$: raw BLE signal set for each device
 $\triangleright R$: emphasized frequency regions
 $\triangleright B$: constructed filter banks
 $\triangleright F$: extracted feature vectors

4.1.1. Pre-Emphasis

Pre-Emphasis is a high-frequency component amplification step. Since high-frequency components in BLE signals generally exhibit smaller magnitudes than low-frequency components, a pre-emphasis operation is applied to balance the spectral distribution and enhance high-frequency details prior to frequency-domain analysis. This operation is implemented using the following pre-emphasis equation:

$$y[n] = x[n] - \alpha \cdot x[n - 1] \quad (1)$$

In Equation (1), $x[n]$ denotes the current signal sample, $x[n - 1]$ represents the previous sample, and α is the pre-emphasis coefficient with a value between 0 and 1. The coefficient α controls the degree of high-frequency amplification, with typical values ranging from 0.95 to 0.97. If α is set too low, high-frequency components are insufficiently emphasized, limiting improvements in spectral contrast. Conversely, excessively large values of α may cause over-amplification of high-frequency noise and introduce numerical instability in subsequent spectral analysis. In the field of speech signal processing, the use of pre-emphasis coefficients in the range of 0.95–0.97 is widely recognized as a common practice. Based on this established convention, the same coefficient range was applied to BLE signals, and experimental results indicate that it is also effective in emphasizing discriminative spectral characteristics in BLE transmissions.

To further examine the influence of this coefficient, we conducted a sensitivity analysis by varying α from 0.90 to 0.99. For each α value, the feature set was regenerated from the raw-signal stage by reapplying the same preprocessing, emphasized-region identification, and filter-bank construction procedures. Except for the value of α , all preprocessing and feature-generation steps were kept identical to the main pipeline, including framing, windowing, FFT, emphasized-region selection, emphasized filter-bank generation, model training, threshold determination, and frame- and signal-level evaluation. This analysis was used to assess whether the selected coefficient substantially affects the emphasized spectral representation and the final authentication performance.

By accentuating high-frequency components while relatively attenuating low-frequency components, pre-emphasis increases the prominence of subtle signal variations and improves spectral balance. This process reduces dominance of low-frequency energy, mitigates noise effects, and enhances the clarity of spectral features, thereby improving the reliability of subsequent signal analysis. Figure 2 illustrates the frequency-dependent spectral change induced by pre-emphasis, represented as the magnitude difference between the pre-emphasized and original BLE signals. In this figure, the gray curve represents the magnitude difference after pre-emphasis, whereas the black horizontal line indicates the zero-difference reference. Figure 3 provides representative time-domain BLE signal examples before and after applying the pre-emphasis filter.

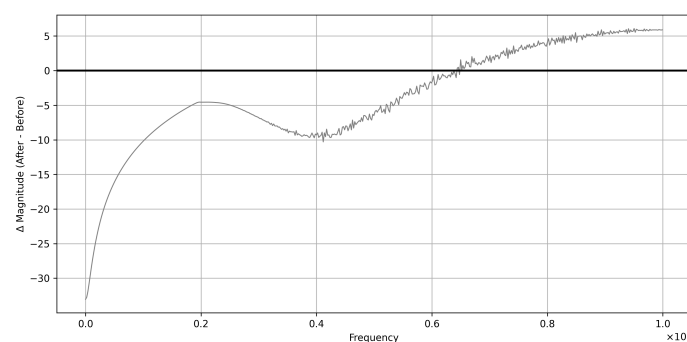


Figure 2. Frequency-dependent spectral change induced by pre-emphasis.

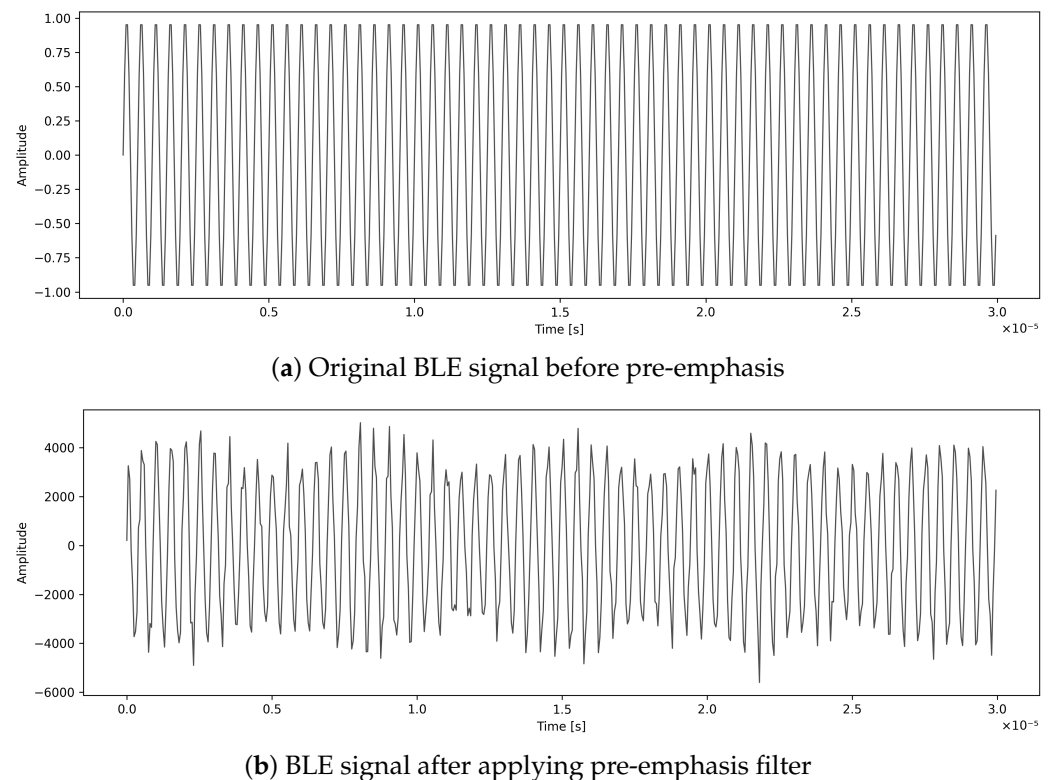


Figure 3. Examples of signals before and after applying pre-emphasis.

BLE signal after pre-emphasis exhibits amplified high-frequency fluctuations compared to the original signal. Before pre-emphasis, the waveform is dominated by smoother, low-frequency components with relatively large amplitudes, which obscure subtle variations embedded in higher-frequency regions. After applying pre-emphasis, rapid signal variations become more pronounced, while the dominance of low-frequency components is reduced.

4.1.2. Framing

Framing is the step of dividing signal data into fixed time segments, ensuring that each frame overlaps partially with adjacent frames. This overlapping design allows frames to retain some of the same information as preceding frames, which helps mitigate abrupt changes in frequency components. This step is particularly crucial when performing transformations such as the Fourier Transform, as it prevents the loss of critical frequency components over time. By analyzing each frame separately while maintaining continuity through overlapping, an approximation of the original signal's frequency characteristics can be obtained.

In our framework, BLE signals are segmented using overlapping frames to capture temporal variations during transmission. In conventional audio signal processing, overlap ratios of approximately 40–50% are widely used, since audio waveforms generally exhibit smooth temporal variations. By contrast, BLE transmissions often involve abrupt and irregular signal changes at short time scales due to packet-level communication and hardware-dependent behaviors. To improve temporal resolution and preserve continuity across adjacent frames, a higher overlap ratio of 60% is adopted. This choice reduces the risk of losing discriminative signal characteristics at frame boundaries and supports more reliable spectral feature extraction in subsequent processing. If the overlap between frames is too low, the ability to capture detailed changes in the signal diminishes, increasing the risk of missing crucial variations, especially in rapidly fluctuating signals.

Additionally, insufficient overlaps may cause the window function to abruptly truncate the signal at the frame's boundary, leading to spectral leakage. Conversely, excessive overlap increases computational complexity, memory usage, and processing time. The resulting frame segmentation is illustrated in Figure 4, showing overlapping frames extracted from the signal.

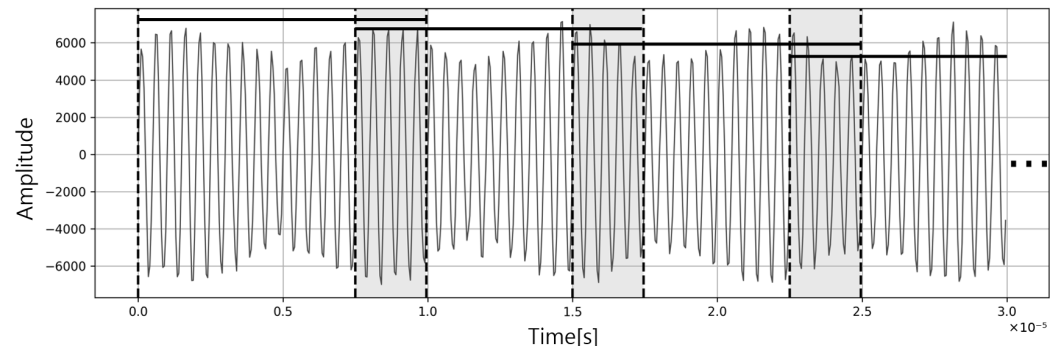


Figure 4. An example of frames of a signal.

4.1.3. Windowing

Windowing is a finite impulse response (FIR) filtering operation that enhances frequency resolution and preserves the original signal form by mitigating discontinuities between frames. Additionally, it helps reduce spectral artifacts that may arise during the Fourier transform due to frame discontinuities. Among various window functions, we employ the Hamming window, which is defined by the following equation:

$$w[n] = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad (2)$$

Here, $w[n]$ denotes the window coefficient at the n -th sample, N represents the total window length, and n is the sample index ranging from 0 to $N-1$. A Hamming window function is applied to each frame to preserve the overall signal shape while smoothing the boundary regions at both ends. As a result, the windowing step improves the quality of the framed signal and facilitates subsequent spectral analysis. An example of the Hamming window is illustrated in Figure 5.

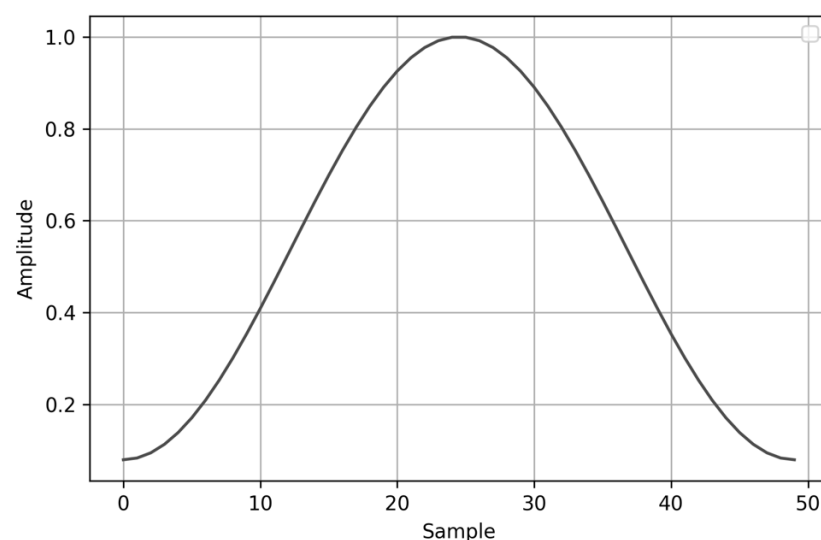


Figure 5. An example of a Hamming window.

The illustration of a specific frame before and after applying the windowing is shown in Figure 6.

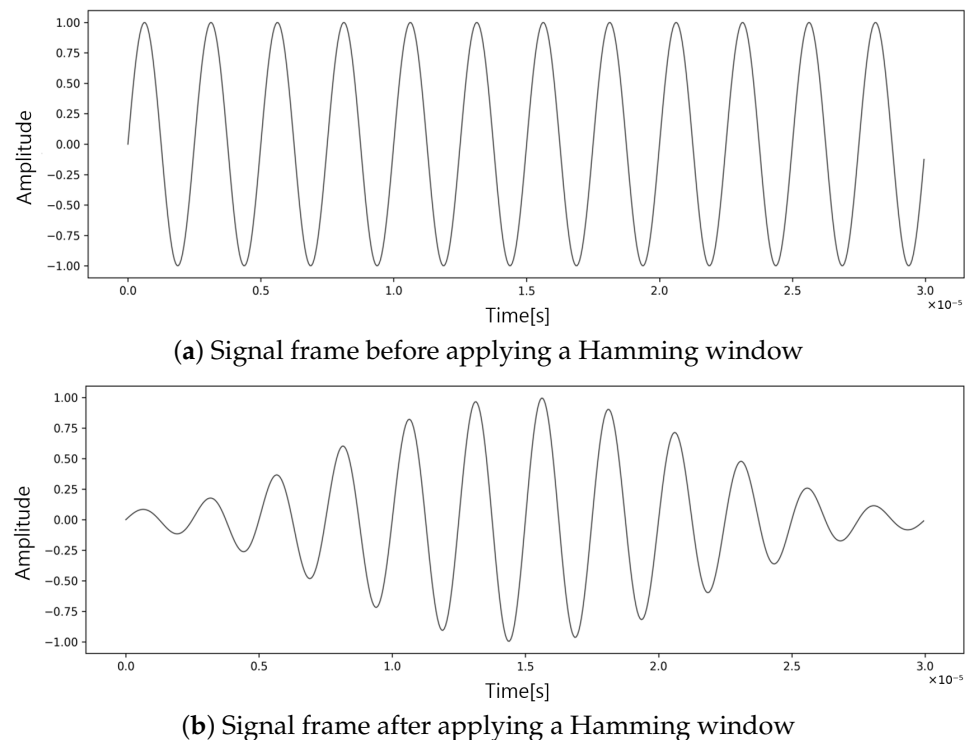


Figure 6. Examples of signals before and after applying a Hamming window.

As shown in Figure 6, applying the Hamming window gradually attenuates the signal amplitude toward the beginning and end of the frame while largely preserving the central portion of the waveform. Prior to windowing, direct segmentation of the signal can introduce abrupt discontinuities at frame boundaries, which may lead to spectral leakage in subsequent frequency-domain analysis. By smoothly tapering the boundary regions, windowing reduces sharp transitions caused by frame truncation and suppresses artificial high-frequency components.

Through this process, the essential characteristics of the signal within the central region of each frame are maintained, while boundary-induced distortions are effectively minimized. As a result, windowing improves the stability and reliability of the spectral representation obtained from FFT-based analysis, thereby enhancing the overall accuracy of subsequent signal processing stages.

4.1.4. FFT

The FFT is a process that converts time-domain data into frequency-domain data using Fourier transform. It is derived from the discrete Fourier transform (DFT) algorithm and is specifically designed to enhance computational efficiency by reducing the complexity of calculations. The mathematical formulation of FFT is as follows:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi}{N}kn} \quad (3)$$

Here, $X[k]$ denotes the k -th frequency component of the resulting spectrum, $x[n]$ represents the n -th time-domain sample of the original signal, and N denotes the total number of samples. The symbol j represents the imaginary unit (i.e., $\sqrt{-1}$), and e denotes the base of the natural logarithm. The index k ranges from 0 to $N - 1$, corresponding to each discrete

frequency component. The FFT result obtained by applying the transformation to the windowed frames is illustrated in Figure 7.

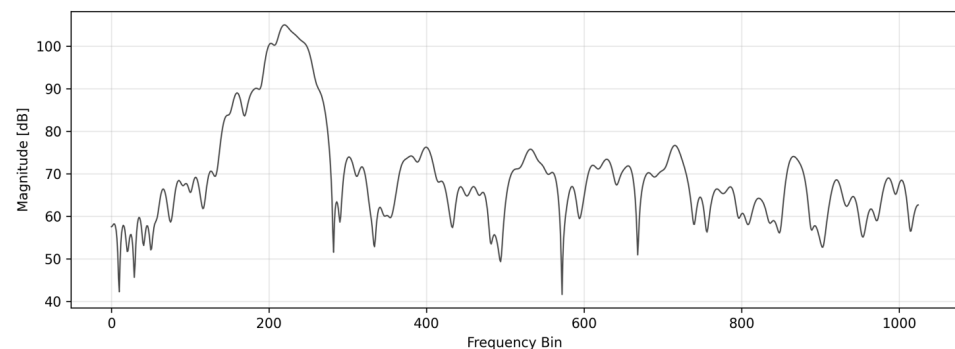


Figure 7. An example of a frame after FFT.

Before applying FFT, the data in the time domain represents amplitude variations over time, with each sample indicating the signal's intensity at a specific time point. After FFT is applied, the transformed data in the frequency domain reflects the strength of each frequency component. By analyzing these results, the proportion and significance of each frequency in the signal can be accurately identified. In this study, FFT is essential, as it enables the identification of frequency ranges where a specific BLE module's signal exhibits distinct differences from other modules, which are then designated as emphasis sections.

Following the FFT transformation, the frequency spectrum is analyzed to determine the frequency ranges where the greatest differences between BLE signal transmission devices occur. The transformed data is divided into sections using a sliding window, and the variance of each section is calculated. Sections with variance values greater than the average are identified as emphasis sections, while the remaining sections are classified as general sections.

For each sliding frequency segment, the variance is computed, and the average variance value is determined. Frequency segments with variance values exceeding the average are designated as emphasis sections to highlight distinguishing characteristics in BLE signal transmission.

4.1.5. Generation and Application of Emphasized Filter Banks

Filter banks are generated and applied to the predefined frequency sections. A linear scale filter bank is used for general sections, while an inverse Mel scale filter bank is applied to emphasis sections. Each frequency section, processed with its respective filter bank, is then combined. Finally, the FFT-transformed data is passed through the fully integrated filter bank to obtain the spectral data.

Generation

In general sections, a linear scale filter bank is applied, providing lower filter density compared to the emphasis sections. This filter bank divides the frequency domain into equal frequency intervals, ensuring that each filter maintains a consistent bandwidth. Since the center frequencies of the filters are evenly spaced, this method offers advantages in capturing the overall characteristics of the signal. The uniform filter spacing across both low- and high-frequency ranges makes it particularly effective for obtaining general spectral features, and its relatively simple design facilitates easy implementation. Therefore, applying the linear scale filter bank to general sections allows for a broad frequency analysis while maintaining a balanced representation of the overall signal characteristics.

For emphasis sections, an inverse Mel scale filter bank is applied, specifically designed for detailed frequency analysis in these regions. Unlike the conventional Mel scale filter

bank, which mimics human auditory perception by being more sensitive to low-frequency components, the inverse Mel scale filter bank provides denser filtering in the high-frequency range. This allows for enhanced sensitivity in detecting high-frequency variations, making it particularly effective for analyzing BLE signals, which predominantly exhibit significant characteristics in the high-frequency domain. Additionally, this filtering approach optimizes resource utilization by allocating higher filter density where it is most needed. By applying the inverse Mel scale filter bank to the emphasis sections, higher-resolution analysis is achieved compared to the general sections.

Finally, the filter banks for the emphasis and general sections are combined to generate a single final filter bank for each module. By applying different filter banks to different sections, this approach ensures a balanced overall signal analysis while allowing high-resolution analysis in critical frequency regions where finer details need to be captured. The structure of the final filter bank is illustrated in Figure 8.

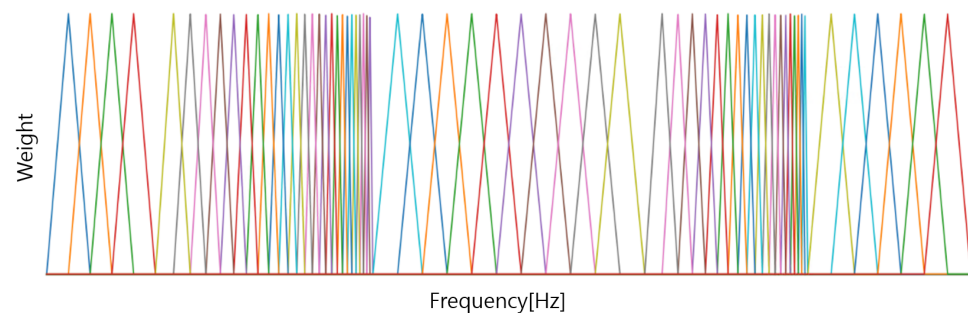


Figure 8. Final filter bank.

As the filter bank weight approaches 0, the proportion of the signal passing through the filter at the corresponding frequency decreases, whereas a weight closer to 1 allows the maximum amount of signal to pass. Accordingly, the final filter bank designed in this study ensures that signals in the emphasis sections pass through with maximum intensity for detailed analysis, while signals in the general sections are comparatively attenuated to maintain a balanced overall analysis.

The algorithm for defining each section, generating the corresponding filter bank, and integrating them into the final filter bank is outlined in Algorithms 2–6.

Algorithm 2 Find Emphasized Regions

<pre> 1: function FINDEMPHASIZEDREGIONS(Φ, W) Window size 2: $R \leftarrow []$ 3: $V \leftarrow []$ 4: $N \leftarrow \text{length}(\Phi) / W$ 5: for $i \leftarrow 1$ to N do 6: $x \leftarrow \Phi[(i-1) \cdot W + 1 : i \cdot W]$ 7: $v \leftarrow \text{Variance}(x)$ 8: $V.\text{append}(v)$ 9: end for 10: $\mu \leftarrow \text{Mean}(V)$ 11: for each $v \in V$ do 12: if $v > \mu$ then 13: $R.\text{append}(v)$ 14: end if 15: end for 16: return R 17: end function </pre>	$\triangleright \Phi$: Frequency-domain features, W: Window size $\triangleright R$: emphasized regions $\triangleright V$: variance values per window $\triangleright N$: number of frequency windows
--	---

Algorithm 3 Generate Emphasized Filters

```

1: function DESIGNINVERSEMELSCALEFILTERS( $r$ )    ▷  $r$ : emphasized frequency regions
2:    $F_e \leftarrow []$                                 ▷  $F_e$ : set of emphasized inverse-Mel filters
3:   for each  $r_i \in r$  do
4:      $f \leftarrow \text{DesignFilterInverseMel}(r_i)$ 
5:      $F_e.\text{append}(f)$ 
6:   end for
7:   return  $F_e$ 
8: end function

```

Algorithm 4 Determine Non-Emphasized Regions

```

1: function GETNONEMPHASIZEDREGIONS( $F, r$ )    ▷  $F$ : frequency set,  $r$ : emphasized
   region set
2:    $R \leftarrow []$                                 ▷  $R$ : set of non-emphasized frequency regions
3:   for each  $f \in F$  do
4:     if  $f \notin r$  then
5:        $R.\text{append}(f)$ 
6:     end if
7:   end for
8:   return  $R$ 
9: end function

```

Algorithm 5 Design Linear Scale Filters

```

1: function DESIGNLINEARSCALEFILTERS( $r$ )    ▷  $r$ : non-emphasized regions
2:    $F \leftarrow []$                                 ▷  $F$ : set of linear-scale filters
3:   for each  $r_i \in r$  do
4:      $f \leftarrow \text{DesignLinearFilter}(r_i)$ 
5:      $F.\text{append}(f)$ 
6:   end for
7:   return  $F$ 
8: end function

```

Algorithm 6 Merge Filters

```

1: function MERGEFILTERS( $E, L$ )    ▷  $E$ : emphasized filters,  $L$ : linear-scale filters
2:    $B \leftarrow E \cup L$                                 ▷  $B$ : merged filter bank
3:   return  $B$ 
4: end function

```

Application

The previously designed emphasized filter bank is applied to the FFT-transformed data to compute the energy of each frequency band and extract spectral data corresponding to the BLE signal transmission device.

Each filter in the filter bank is applied to a specific band of the frequency spectrum, with its weight multiplied by the respective frequency components to calculate the energy within that band. A single filter processes all frequency components within its designated band, and the weighted multiplication results are summed to determine the output energy of the filter.

The values obtained after passing through the filter bank represent the energy distribution across different frequency bands, reflecting the characteristics of each band. The mathematical expressions for computing the weighted product of filter coefficients and frequency components, as well as the summation formula for determining the filter's output energy, are provided in Equations (4) and (5).

$$S_m[k] = |X[k]| \cdot H_m[k] \quad (4)$$

Here, $S_m[k]$ denotes the multiplication result at the k -th frequency component for the m -th filter. The term $|X[k]|$ represents the magnitude of the FFT output at frequency index k , which corresponds to the signal strength at that frequency. The function $H_m[k]$ denotes the weighting coefficient of the m -th filter at the k -th frequency component.

$$E_m = \sum_{k=0}^{N-1} S_m[k] = \sum_{k=0}^{N-1} |X[k]| \cdot H_m[k] \quad (5)$$

Here, E_m denotes the output energy of the m -th filter. The variable N represents the total number of frequency components in the FFT result, and $S_m[k]$ corresponds to the value computed in Equation (4), which, as described above, represents the multiplication result at the k -th frequency component for the m -th filter.

Ultimately, each extracted output energy serves as a representative vector characterizing the signal within a specific frequency range. The collection of all vectors computed within a filter bank forms the spectral data, which is utilized as the feature information in this study. The algorithm for applying the filter bank to extract spectral data is provided in Algorithm 7.

Algorithm 7 Apply Filter Bank

<pre> 1: function APPLYFILTERBANK(B, X) 2: $C \leftarrow []$ 3: for each $x \in X$ do 4: $c \leftarrow \text{Apply}(B, x)$ 5: $C.\text{append}(c)$ 6: end for 7: return C 8: end function </pre>	▷ B: filter bank, X: FFT frames ▷ C: cepstral coefficient set
--	--

4.2. Threshold Setting and Device Authentication

4.2.1. Autoencoder-DNN-Based Model

To identify normal and abnormal BLE signals, a DNN algorithm with an autoencoder structure is employed. An autoencoder is an unsupervised learning algorithm that compresses input data into a low-dimensional vector and subsequently reconstructs it back into its original high-dimensional form. The unsupervised learning approach is particularly effective for training on unlabeled data, making it well-suited for BLE signals, which typically generate vast amounts of data, with most real-world data being normal. By leveraging an autoencoder, the model can effectively learn the patterns of BLE signals.

Furthermore, during the process of compressing the input data into a lower-dimensional representation and then reconstructing it, key features of the signal are extracted while unnecessary noise is filtered out. BLE signals are often affected by various environmental factors that introduce noise, but this algorithm helps mitigate noise and extract essential signal characteristics. Additionally, since normal signals exhibit a low reconstruction error while abnormal signals have a significantly higher reconstruction error, this property is utilized for BLE anomaly detection in this study.

Moreover, the Autoencoder-DNN algorithm demonstrates strong learning capabilities for large datasets and effectively captures features in BLE signals, which exhibit inherent variability. The parameter settings used in this algorithm are presented in Table 1.

Table 1 summarizes the model configuration used for the Autoencoder-DNN. The number of fully connected and batch-normalization layers defines the depth of the reconstruction model, while the bottleneck layer controls the degree of feature compression. The exponential linear unit (ELU) activation and adaptive moment estimation (Adam) optimizer were adopted to support stable nonlinear learning, and the mean squared error

(MSE) loss was used because the authentication score is based on reconstruction error. The batch size, learning rate, number of epochs, and Elastic Net weight were fixed across experiments to ensure consistent comparison among feature representations.

Table 1. Parameter configuration of the Autoencoder-DNN model.

Parameter	Value/Configuration
Number of Layers	26 layers: 13 FC + 13 Batch Norm
Bottleneck Structure	FC Layer with 2 neurons
Activation Function	ELU
Loss Function	MSE
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epochs	1000
Elastic Net (L1/L2 Weight)	0.01

Network Architecture

The algorithm employed in this study consists of 13 fully connected layers, each followed by a batch normalization layer, resulting in a total of 26 layers. A multilayer neural network architecture is adopted to effectively learn complex nonlinear patterns, while batch normalization layers are incorporated to enhance training stability and accelerate the learning process. Additionally, a bottleneck layer with two neurons is utilized in the autoencoder to efficiently compress the essential features of the input data, enabling accurate reconstruction based on the extracted representations.

Activation Function

The ELU function is employed as the activation function in this study. The ELU function addresses the limitations of the rectified linear unit (ReLU) by providing nonzero outputs for negative input values, thereby mitigating the vanishing gradient problem. In addition, ELU encourages the mean activation of neurons to approach zero during the early stages of training, which contributes to faster convergence. The ELU function is defined as follows:

$$\text{ELU}(x) = \begin{cases} x, & \text{if } x > 0, \\ \alpha(e^x - 1), & \text{if } x \leq 0, \end{cases} \quad (6)$$

where x denotes the input value and α is a positive hyperparameter that controls the saturation level for negative inputs.

Loss Function

The loss function used in this study is the MSE function. The MSE loss function is effective in minimizing reconstruction errors by squaring and averaging the differences between the input data and the reconstructed data, thereby enhancing reconstruction accuracy. The MSE is defined as:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (7)$$

where n denotes the number of data samples, y_i represents the ground-truth value, and $\hat{y}_i$ denotes the corresponding predicted value.

Optimization Algorithm

The Adam optimizer is used as the optimization algorithm. By adopting the default learning rate of 0.001, the model achieves stable and fast convergence across various problems. The gradient of the objective function with respect to the model parameters at iteration t is computed as:

$$g_t = \nabla_{\theta} J(\theta_{t-1}) \quad (8)$$

where $J(\cdot)$ denotes the loss function and θ_{t-1} represents the model parameters from the previous iteration. The first-order moment estimate is updated as:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (9)$$

and the second-order moment estimate is updated as:

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (10)$$

where m_t and v_t denote the estimates of the first and second moments of the gradient, respectively. To compensate for the bias introduced during initialization, bias-corrected moment estimates are computed as:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (11)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (12)$$

Finally, the model parameters are updated according to:

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} \quad (13)$$

where θ_t denotes the updated model parameters, η is the learning rate, g_t is the gradient at iteration t , and β_1 and β_2 represent the exponential decay rates for the moment estimates. The constant ε is a small positive value added to ensure numerical stability.

Normalization

Elastic Net regularization, which combines L1 (Lasso) and L2 (Ridge) regularization techniques, is used for normalization. This approach simultaneously performs feature selection and weight shrinkage, effectively controlling model complexity, preventing overfitting, and enhancing generalization performance. The strengths of L1 and L2 regularization are both set to 0.01 to maintain an appropriate balance and maximize the regularization effect. The Elastic Net regularization is defined as:

$$L(\beta) = \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 \quad (14)$$

where β denotes the regression coefficient vector, X represents the input data matrix, and y denotes the ground-truth output vector. The parameters λ_1 and λ_2 are the regularization weights for the L_1 and L_2 penalties, respectively. The term $\|\beta\|_1$ represents the sum of the absolute values of the coefficients, while $\|\beta\|_2$ denotes the Euclidean norm of the coefficient vector.

Batch normalization is employed to normalize the inputs of each layer, thereby alleviating the vanishing gradient problem and improving training stability.

4.2.2. Reconstruction Error

The reconstruction error is calculated by measuring the difference between the input data and the reconstructed data by the autoencoder. This process involves computing the MSE, which plays a crucial role in determining signal anomalies. The reconstruction error is defined as:

$$\text{Reconstruction Error} = \text{MSE}(x, x') \quad (15)$$

where x denotes the original signal and x' represents the reconstructed signal.

4.2.3. Threshold Setting

The method for setting the threshold to distinguish between normal and abnormal BLE signals based on the extracted MSE values consists of the following steps: data partitioning, MSE range segmentation, candidate threshold selection, F1-score-based evaluation, and final threshold determination. The complete dataset obtained during the feature extraction process is divided into training data for the Autoencoder-DNN algorithm and a separate dataset for threshold setting. After computing the reconstruction error for the signals to be classified using the trained Autoencoder-DNN algorithm, the corresponding MSE values are calculated. The computed MSE values are then sorted in ascending order and partitioned into multiple intervals, from which representative midpoint values are selected as candidate thresholds. Each candidate threshold is evaluated by classifying BLE signal transmission devices and assessing performance using the F1-score. The candidate that yields the highest F1-score is selected as the final threshold. The overall algorithm for threshold setting is presented in Algorithm 8.

Because this F1-score-based threshold uses labeled calibration data, we evaluated normal-only statistical thresholding strategies to examine threshold generalization. In these experiments, abnormal samples and abnormal labels were not used for threshold determination. The compared normal-only methods included the 99th percentile of normal reconstruction errors, mean plus three standard deviations, and median plus three scaled median absolute deviations. These thresholding strategies were used to assess whether the authentication framework can be operated without relying on labeled abnormal calibration data.

Algorithm 8 Set Threshold

```

1: function SETTHRESHOLD( $F$ )                                      $\triangleright F$ : feature vectors
2:    $(T, U) \leftarrow \text{Split}(F, 0.7)$                               $\triangleright T$ : training data,  $U$ : validation data
3:    $M \leftarrow \text{TrainAutoencoder}(T)$ 
4:    $E \leftarrow []$                                                $\triangleright E$ : reconstruction error (MSE) set
5:   for each  $u \in U$  do
6:      $o \leftarrow M(u)$ 
7:      $e \leftarrow \text{MSE}(u, o)$ 
8:      $E.\text{append}(e)$ 
9:   end for
10:   $\theta \leftarrow \text{SelectThreshold}(E)$ 
11:  return  $\theta$ 
12: end function

```

The determine-optimal-threshold algorithm, which establishes the final threshold, is detailed in Algorithm 9.

Algorithm 9 Determine Optimal Threshold

```

1: function DETERMINEOPTIMALTHRESHOLD( $E$ )  $\triangleright E$ : MSE/reconstruction error values
2:    $S \leftarrow \text{Sort}(E)$   $\triangleright S$ : sorted reconstruction errors (ascending)
3:    $N \leftarrow 100$   $\triangleright N$ : number of intervals
4:    $I \leftarrow \text{length}(S)/N$ 
5:    $C \leftarrow []$   $\triangleright C$ : candidate threshold set
6:   for  $i \leftarrow 1$  to  $N - 1$  do
7:      $c \leftarrow (S[i \cdot I] + S[(i+1) \cdot I])/2$ 
8:      $C.\text{append}(c)$ 
9:   end for
10:   $\theta \leftarrow \text{None}; f \leftarrow 0$   $\triangleright \theta$ : optimal threshold;  $f$ : best F1-score
11:  for each  $c \in C$  do
12:     $g \leftarrow \text{F1}(c)$ 
13:    if  $g \geq f$  then
14:       $f \leftarrow g; \theta \leftarrow c$ 
15:    end if
16:  end for
17:  return  $\theta$ 
18: end function

```

4.2.4. Final Decision

The MSE value of the BLE signal from the device to be authenticated is calculated and compared with the previously determined threshold to detect anomalies and perform device authentication. The complete algorithm for signal authentication is presented in Algorithm 10.

Algorithm 10 Signal Identification

```

1: function IDENTIFYSIGNALS( $S, B, \theta, M, D$ )  $\triangleright S$ : signals,  $B$ : filter banks,  $\theta$ : threshold,  $M$ :
   trained model,  $D$ : device set
2:    $I \leftarrow \{\}$   $\triangleright I$ : identified signal results per device
3:   for each  $d \in D$  do
4:      $s \leftarrow S[d]$ 
5:      $s \leftarrow \text{PreEmphasis}(s)$ 
6:      $f \leftarrow \text{Framing}(s)$ 
7:      $w \leftarrow \text{Windowing}(f)$ 
8:      $X \leftarrow \text{FFT}(w)$ 
9:      $b \leftarrow B[d]$ 
10:     $c \leftarrow \text{ApplyFilterBank}(b, X)$   $\triangleright c$ : cepstral coefficients
11:     $o \leftarrow M(c)$   $\triangleright o$ : reconstructed output
12:     $e \leftarrow \text{MSE}(c, o)$   $\triangleright e$ : reconstruction error
13:    if  $e < \theta$  then
14:       $I[d] \leftarrow \text{true}$ 
15:    else
16:       $I[d] \leftarrow \text{false}$ 
17:    end if
18:  end for
19:  return  $I$ 
20: end function

```

5. Evaluation

In this section, we present the experiments and results used to validate the proposed framework. The following subsections describe the comparative settings used for validation, including datasets constructed with different feature-processing methods, alternative definitions of emphasis regions, and baseline machine learning algorithms. We then report performance metrics under varying parameters and analyze how the metrics change. Using

these settings, we performed comparative experiments on raw signal data and on data processed with conventional filtering.

To support this evaluation, we collected signal data from 40 BLE transmitters, with 500 samples per transmitter. Figure 9 shows our RF measurement setup for BLE signal acquisition, including the antenna placement and the receiver configuration. We used a single Arduino Uno board sequentially equipped with forty BLE transmitting modules configured as BLE advertising transmitters, and a HackRF One with a 2.4 GHz RF antenna to capture signals in the 2.4 GHz ISM band. The BLE transmitting modules periodically broadcasted BLE advertising packets with different device addresses and payload patterns to emulate multiple legitimate devices in a realistic wireless environment. The evaluation follows a device-specific authentication setting rather than a closed-set multi-class identification setting: for each enrolled transmitter, a normal-device model is trained, and an incoming BLE advertising signal is evaluated against the corresponding enrolled-device model to determine whether it should be accepted as legitimate or rejected as anomalous. For RF acquisition, we set the sampling rate to 20 MHz and captured continuous time-domain BLE baseband signals by tuning the receiver to the BLE advertising band. The received signals were recorded as raw waveform data and subsequently processed through framing and spectral analysis for further signal processing and evaluation.

The acquisition parameters were selected to provide a controlled and repeatable BLE RF fingerprinting scenario. The 20 MHz sampling rate provides sufficient temporal resolution for capturing BLE advertising signals around the 2.4 GHz ISM band using the HackRF One receiver, while the same antenna and receiver configuration were maintained for all transmitters to ensure fair comparison. Because BLE advertising signals are based on Gaussian frequency-shift keying (GFSK) modulation rather than orthogonal frequency-division multiplexing (OFDM), the number of subcarriers is not a configurable scenario parameter in this experimental setting. Instead, the main controllable RF factors include receiver hardware, antenna placement, capture bandwidth, filtering configuration, and channel conditions. These factors may affect absolute performance in deployment, but they were fixed in this study to isolate the effect of the proposed feature representation.

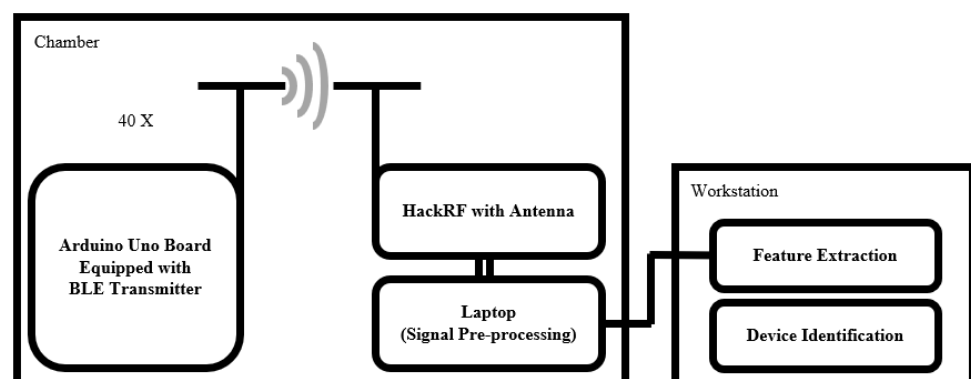


Figure 9. Experimental setup for BLE signal acquisition.

5.1. Experimental Setting

5.1.1. Datasets Used in Experiments

Signals were recorded from 40 BLE transmitters (normal devices), with 500 signals collected per device, and each signal was stored as an individual file. The collected signals were processed using the procedure described in Section 4.1. Each signal was segmented into 73 frames, and each frame was represented by 2048 frequency components after spectral transformation. This resulted in a total of 1,460,000 frame-level instances from

normal devices ($40 \times 500 \times 73$ frames). In the experiments, each frame was treated as an individual training instance for model development and evaluation.

The 40 normal devices were used to construct the normal dataset, while a separate set of 10 unseen devices, not used for training, was exclusively reserved for generating anomalous samples during evaluation. For each normal device, the resulting 36,500 frames (500×73) were divided into training, test, and threshold-selection subsets in a 7:1.5:1.5 ratio. Consequently, per normal device, 25,550 frames were allocated to the training set, 5475 frames were allocated to the test set, and 5475 frames were allocated to the threshold-selection set. Abnormal data were derived solely from the pool of 10 unseen devices and were not included in training. For each enrolled-device evaluation, 3650 abnormal frames were sampled from the unseen-device pool for the test set, and another 3650 abnormal frames were sampled for the threshold-selection set. Thus, each test and threshold-selection split contained 5475 normal frames and 3650 abnormal frames, corresponding to a fixed frame-level normal-to-abnormal ratio of 6:4.

Although frame-level evaluation treats each frame as an individual instance, device authentication is ultimately performed at the signal level. Therefore, we conducted a signal-level evaluation by grouping the frame-level scores belonging to the same BLE signal according to the preserved signal grouping used for evaluation. Let F denote the number of frame-level scores assigned to a signal, and let c_s denote the number of frames classified as normal in signal s . For each signal, frame-level normal/abnormal decisions were first obtained using the selected threshold, and the signal was accepted as normal when $c_s \geq n$. We compared three aggregation-threshold selection rules. In the majority rule, $n = \lceil F/2 \rceil$. In the labeled calibration rule, n was selected from the candidate set $\{0, \dots, F\}$ to maximize the normal-class F1-score on the calibration split. In the normal-only rule, n was selected using only normal calibration signals to satisfy a 99% normal-signal acceptance criterion. The selected aggregation rule was then applied unchanged to the held-out test split.

Using this commonly processed dataset as input, multiple dataset variants were constructed to investigate how different frequency representations and emphasis strategies affect performance. Four types of datasets were used in the experiments: Raw Spectrum Data, Uniform Filter-Bank Spectrum Data, Mean-Difference-Emphasized Spectrum Data, and Window-Variance-Emphasized Spectrum Data. These datasets differ in how emphasized frequency intervals are selected and how filter banks are applied.

These benchmark feature sets were selected to isolate the contribution of each design component. Raw Spectrum Data serve as a non-filter-bank baseline that preserves the direct FFT representation. Uniform Filter-Bank Spectrum Data evaluate the effect of applying conventional filter banks without frequency emphasis. Mean-Difference-Emphasized Spectrum Data and Window-Variance-Emphasized Spectrum Data evaluate whether selecting device-discriminative spectral intervals improves authentication performance. Therefore, the benchmark design provides an ablation-oriented comparison between no filtering, uniform filtering, and the proposed emphasis-based filtering strategies.

Raw Spectrum Data were obtained by applying the FFT as described at the end of Section 4.1.4, without normalization or frequency emphasis. Uniform Filter-Bank Spectrum Data were generated by applying conventional filter banks to the raw spectrum without emphasizing any particular frequency interval. In this case, four filter-bank types (Linear Scale, Gammatone Scale, Mel Scale, and Inverse Mel Scale) were applied uniformly across the full frequency range.

Mean-Difference-Emphasized Spectrum Data and Window-Variance-Emphasized Spectrum Data were generated using the proposed framework. After transforming signals into the frequency domain, we computed cepstral coefficients and used their amplitudes to determine emphasized frequency intervals. Unlike the uniform baseline, these emphasized

datasets applied filter banks only within the selected intervals and compared performance across different partitioning strategies. For both emphasized datasets, the Linear, Gammatone, Mel, and Inverse Mel filter banks were applied to the identified emphasis intervals for comparative evaluation.

Mean-Difference-Emphasized Spectrum Data selected intervals where the mean cepstral coefficient amplitudes exhibited larger inter-device differences across BLE transmitters, capturing static magnitude-based distinctions. Window-Variance-Emphasized Spectrum Data selected intervals by sliding a window over the cepstral coefficient amplitudes, computing the variance within each window, and choosing windows with higher variance, thereby capturing variability-based distinctions rather than absolute magnitudes.

The four filter banks represent different frequency partitioning strategies within the selected intervals, and their shapes and frequency responses are illustrated in Figure 10. The Linear Scale filter bank divides the frequency range into equal intervals with constant bandwidth. The Gammatone Scale filter bank emulates cochlear frequency selectivity and provides higher resolution at lower frequencies. The Mel Scale filter bank applies perceptual, nonlinear partitioning with denser allocation at lower frequencies and sparser allocation at higher frequencies. The Inverse Mel Scale filter bank reverses this allocation by placing denser filters at higher frequencies, which can be useful for capturing subtle high-frequency variations related to hardware-dependent characteristics of BLE transmitters within the emphasized intervals.

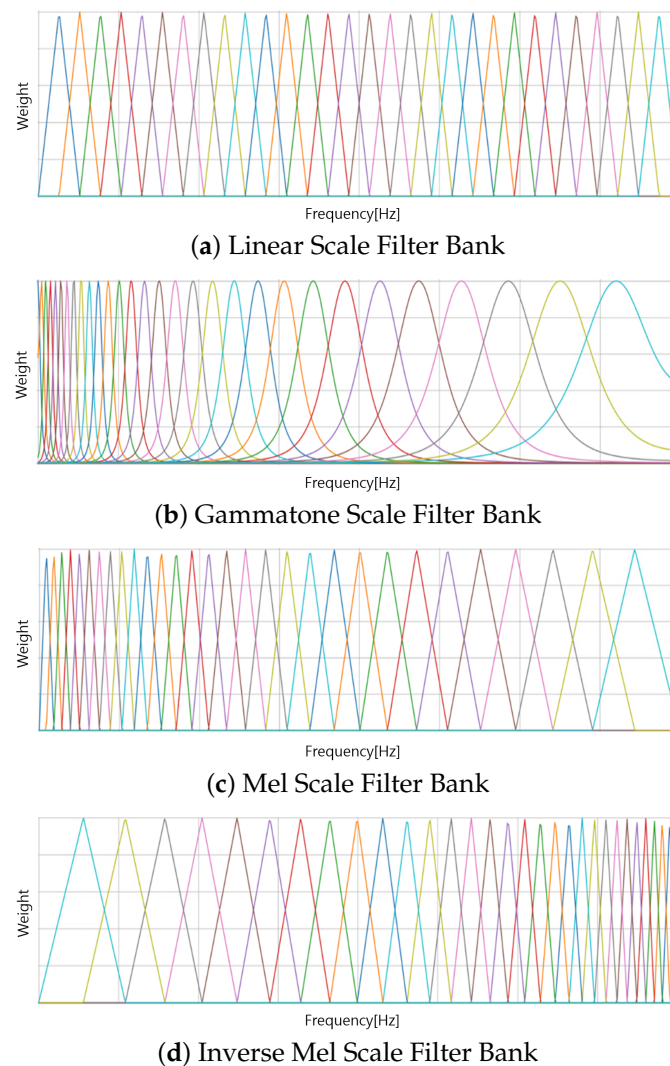


Figure 10. Examples of conventional filter banks.

These four filter-bank scales were included as benchmark alternatives because they represent complementary assumptions about frequency partitioning. Linear Scale provides an equal-bandwidth engineering baseline, Mel Scale provides a widely used nonlinear spectral baseline, Gammatone Scale provides a cochlear-inspired auditory filter-bank baseline, and Inverse Mel Scale tests whether denser high-frequency resolution is beneficial for BLE hardware-dependent spectral variations. Including these alternatives prevents the comparison from depending on a single filter-bank scale and clarifies whether the proposed emphasis strategy remains effective across different spectral partitioning choices.

5.1.2. Machine Learning Algorithms Used in Experiments

A total of five algorithms were evaluated in comparative experiments using combinations of the four dataset types and the four filter-bank types described above. The evaluated methods include Autoencoder-DNN, Autoencoder-1D CNN, One-Class SVM (OCSVM), Isolation Forest (IF), and Local Outlier Factor (LOF). These baseline models were selected because practical RF fingerprint-based authentication typically requires device-specific modeling, and collecting labeled data for all possible devices and attack conditions is not realistic in real deployments. For this reason, we focus on unsupervised anomaly detection methods that can be trained using only legitimate signals from a target device and can flag deviating signals as anomalies. This selection spans complementary unsupervised paradigms, including reconstruction-based deep models, boundary learning, ensemble isolation, and density-based outlier scoring, enabling a robust evaluation under practical constraints.

Autoencoder-DNN uses a deep neural network-based autoencoder to learn compact latent representations and reconstruct the input, which helps capture complex nonlinear patterns for anomaly detection. Autoencoder-1D CNN incorporates one-dimensional convolutional layers within an autoencoder, enabling effective extraction of local sequential patterns and supporting robust anomaly detection. OCSVM is an unsupervised approach that learns a decision boundary from normal data and flags samples outside the boundary as anomalies. Isolation Forest detects anomalies by isolating data points through random partitioning, without assuming any specific data distribution. LOF identifies outliers by comparing the local density of each sample with that of its neighbors, labeling samples in relatively sparse regions as anomalies.

The inclusion of these five algorithms allows the evaluation to cover both deep reconstruction-based authentication and classical anomaly-detection paradigms. AE-DNN and AE-1D CNN test whether nonlinear reconstruction models can learn normal BLE fingerprint distributions from emphasized spectral features. OCSVM provides a boundary-based one-class baseline, Isolation Forest evaluates isolation-based anomaly scoring, and LOF represents density-based local outlier detection. This combination enables the proposed feature representations to be assessed across heterogeneous learning assumptions rather than under a single model family.

The five algorithms used in the evaluation follow the methodological definitions established in the preceding sections. The AE-DNN architecture, including its layer depth, bottleneck compression, activation function, optimizer, loss function, and training parameters, is specified in Section 4.2.1, while the reconstruction-error-based authentication rule and thresholding procedure are described in Section 4.2.3. AE-1D CNN is evaluated under the same reconstruction-error authentication protocol, but replaces the fully connected encoder–decoder structure with one-dimensional convolutional layers to examine local spectral-pattern learning. OCSVM, IF, and LOF use the same emphasized spectral features described in Section 4.1 and the same normal-device training scenario described in

Section 5.1.1. Therefore, the algorithm setup isolates the effect of the learning paradigm while keeping the feature-generation and authentication pipeline consistent.

5.2. Experimental Results and Analysis

Experiments were conducted by combining four data types, four filter-bank types, and five algorithms under different emphasis-region selection methods. A total of 80 experimental combinations were derived from the four data types, five algorithms, and four filter bank types. However, since the Raw Spectrum Data does not employ filter banks, experiments were conducted on 65 combinations in total.

Precision, Recall, and F1-score were used as evaluation metrics and were computed from the confusion matrix elements TP, TN, FP, and FN. In the context of RF fingerprint-based authentication, each test sample is either accepted as legitimate or rejected as anomalous. A true positive (TP) corresponds to a legitimate signal from the enrolled transmitter that is correctly accepted. A true negative (TN) corresponds to a non-enrolled or anomalous signal that is correctly rejected. A false positive (FP) corresponds to a non-enrolled or anomalous signal that is incorrectly accepted as legitimate, resulting in unauthorized acceptance. A false negative (FN) corresponds to a legitimate signal that is incorrectly rejected, resulting in erroneous rejection of an authorized transmitter.

Precision represents the proportion of correctly accepted legitimate samples among all samples accepted as legitimate, while Recall denotes the proportion of legitimate samples that are correctly accepted. The F1-score is the harmonic mean of Precision and Recall, providing a balanced assessment of the two measures. The corresponding equations are defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (16)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (17)$$

$$\text{F1-score} = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}} \quad (18)$$

In Equations (16)–(18), TP, TN, FP, and FN denote the numbers of true positives, true negatives, false positives, and false negatives, respectively.

The complete set of experimental results is summarized in Table A1. Figure 11a–c present the Precision, Recall, and F1-score values for all 65 combinations. Table A1 contains the results grouped by dataset type. In addition, Figure 11a–c summarize the detection performance for each combination of feature representation and filter-bank scale, evaluated under five algorithms. The x-axis labels follow the shorthand Feature Type-Filter Bank Type to keep the plots readable. UFB denotes *Uniform Filter-Bank Spectrum Data* features without emphasis, MD-Emph denotes *Mean-Difference-Emphasized Spectrum Data* features where salient regions are highlighted based on mean-difference criteria, and WV-Emph denotes *Window-Variance-Emphasized Spectrum Data* features where emphasis is applied according to window-level variance. For the filter-bank configuration, Mel, iMel, Gam, and Lin indicate Mel-scale, inverse-Mel-scale, gammatone-scale, and linear-scale filter banks, respectively. Within each category, the five bars correspond to IF, OCSVM, LOF, AE-1D CNN, and AE-DNN, enabling a direct comparison of how the same feature representation behaves under different learning paradigms.

The best-performing and main proposed configuration was obtained with Mean-Difference-Emphasized Spectrum Data, Autoencoder-DNN, and the Inverse Mel Scale filter bank applied within the selected emphasis intervals, achieving Precision 0.9948, Recall 0.9901, and F1-score 0.9927. Mean-Difference-Emphasized Spectrum Data select emphasis intervals based on inter-device differences in mean spectral values, and the

resulting performance clearly exceeded the baseline representations without emphasis. For example, when the same Autoencoder-DNN was applied to Raw Spectrum Data, the performance was Precision 0.6934, Recall 0.6067, and F1-score 0.6472, indicating an F1-score improvement of 0.3455 when emphasis-based features were used. Even compared with the non-emphasized filter-bank baseline using the same algorithm and filter bank, namely Uniform Filter-Bank Spectrum Data with Autoencoder-DNN and Inverse Mel Scale, which achieved Precision 0.9091, Recall 0.8449, and F1-score 0.8758, the emphasized configuration provided a substantial gain. The near-saturated Precision and Recall values in the best case indicate that the proposed MD-Emph inverse-Mel representation helps the model simultaneously reduce unauthorized acceptance and erroneous rejection. In addition to this peak configuration, the aggregate trends show that Window-Variance-Emphasized Spectrum Data can provide a more stable feature-generation alternative across broad experimental variations. Therefore, WV-Emph is discussed as a robustness-oriented alternative when generalization across feature-generation conditions is prioritized, while the main proposed configuration remains MD-Emph with the Inverse Mel Scale filter bank and Autoencoder-DNN.

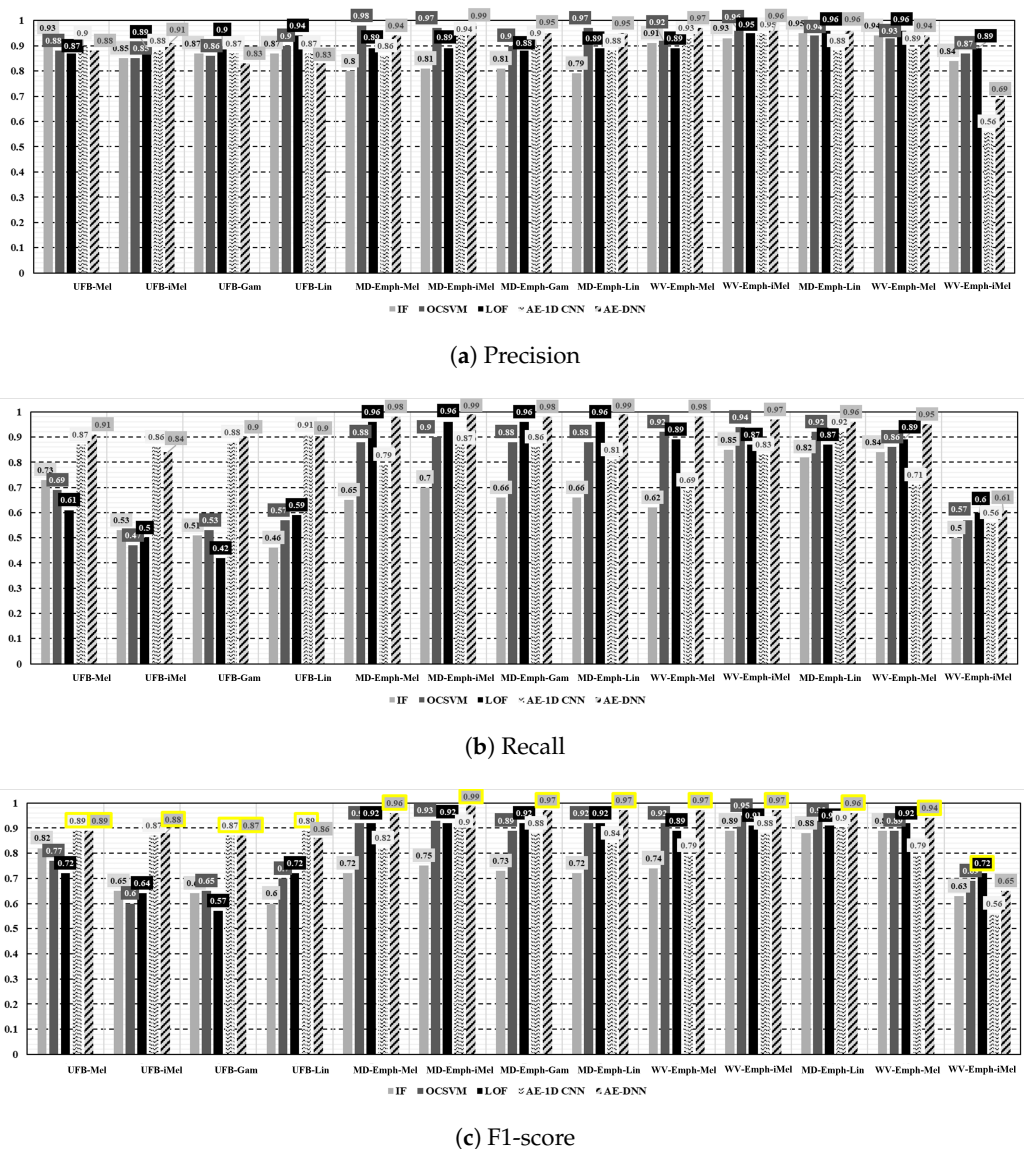


Figure 11. Precision (a), Recall (b), and F1-score (c) for all 65 experimental combinations.

In the following subsections, the results are organized and compared from three perspectives. First, performance is analyzed by dataset type. Next, results are compared by filter-bank type. Finally, the performance of the machine learning algorithms is examined in detail.

5.2.1. Performance Comparison Among Different Feature Processing Methods

Figure 11a–c and Table 2 show a consistent advantage for emphasis-based feature processing over the non-emphasized baselines. In Figure 11c, the highest F1-score outcomes appear predominantly under the emphasized spectrum datasets, while weaker outcomes are more common in the Raw and uniform-processing settings where Recall drops visibly in Figure 11b. Table 2 quantifies this overall trend: Window-Variance-Emphasized Spectrum Data achieves the highest mean F1-score, 0.8871 ± 0.0597 , and Mean-Difference-Emphasized Spectrum Data follows with 0.8801 ± 0.0858 . In contrast, Uniform Filter-Bank Spectrum Data and Raw Spectrum Data are substantially lower, with 0.7687 ± 0.1141 and 0.6534 ± 0.0547 , respectively. The min–max ranges computed from the full table further clarify what Figure 11c suggests: Window-Variance-Emphasized Spectrum Data maintains a comparatively strong performance floor, with F1-score ranging from 0.7352 to 0.9740, whereas Uniform Filter-Bank Spectrum Data ranges from 0.5684 to 0.8930 and Raw Spectrum Data ranges from 0.5572 to 0.7175.

The Recall patterns in Figure 11b explain much of the F1-score separation. Raw Spectrum Data shows a low mean Recall, 0.5672 ± 0.0374 , while Uniform Filter-Bank Spectrum Data exhibits both low and highly variable Recall, 0.6832 ± 0.1788 . The full-table ranges reinforce the worst-case mechanisms: Uniform Filter-Bank Spectrum Data reaches a minimum Recall of 0.4163, while Window-Variance-Emphasized Spectrum Data maintains a higher minimum Recall of 0.6163 and reaches a maximum Recall of 0.9753. Precision in Figure 11a is generally not the limiting factor for the emphasized datasets, consistent with Table 2 where emphasis-based methods keep high mean Precision. Window-Variance-Emphasized Spectrum Data achieves mean Precision 0.9309 ± 0.0263 , and its Precision ranges from 0.8755 to 0.9719, indicating strong stability. By contrast, Raw Spectrum Data exhibits both lower mean Precision, 0.7705 ± 0.1267 , and a wide Precision range from 0.5578 to 0.8895, which aligns with the unstable Precision behavior seen in Figure 11a. Overall, Figure 11a–c and Table 2 indicate that emphasis-based processing improves F1-score mainly by preventing Recall degradation while maintaining high Precision, and Window-Variance-Emphasized Spectrum Data provides the most robust performance envelope across configurations.

Table 2. Results by data type.

Data Type	Precision (Mean $\pm$ Std)	Recall (Mean $\pm$ Std)	F1-Score (Mean $\pm$ Std)
Uniform Filter-Bank Spectrum	0.8787 ± 0.0279	0.6832 ± 0.1788	0.7687 ± 0.1141
Mean-Diff.-Emphasized Spectrum	0.8916 ± 0.0613	0.8688 ± 0.1151	0.8801 ± 0.0858
Win.-Var.-Emphasized Spectrum	0.9309 ± 0.0263	0.8473 ± 0.0936	0.8871 ± 0.0597
Raw Spectrum Data	0.7705 ± 0.1267	0.5672 ± 0.0374	0.6534 ± 0.0547

5.2.2. Performance Comparison Among Different Filter Bank Methods

The filter-bank effects visible in Figure 11a–c are consistent with Table 3, but the key point is that the filter bank should be interpreted jointly with robustness, not only with peak or average performance. Table 3 reports the highest mean F1-score for Inverse Mel Scale, 0.9054 ± 0.1215 , followed by Gammatone Scale, 0.8934 ± 0.1217 , and Linear Scale, 0.8806 ± 0.1049 . Mel Scale shows mean F1-score 0.8662 with a smaller standard deviation

of 0.0845. The full-table ranges explain the figure-level variability: Inverse Mel Scale spans a very wide F1-score range from 0.6013 to 0.9927, meaning it can deliver the best-case F1-score but also produces much weaker outcomes depending on the configuration. In contrast, Mel Scale maintains a higher lower bound, with F1-score ranging from 0.7169 to 0.9740, which matches the more consistently strong behavior seen across many settings in Figure 11c.

A similar trade-off appears in Recall, which is critical for stable F1-score behavior. Inverse Mel Scale has the highest mean Recall, 0.8843 ± 0.1681 , and reaches a maximum Recall of 0.9901. However, its minimum Recall drops to 0.4664, showing strong configuration dependence that is also reflected in Figure 11b. Mel Scale yields a more conservative but stable envelope: mean Recall is 0.8316 ± 0.1295 , and the Recall range is 0.6119 to 0.9828, supporting robust performance across diverse conditions. Precision differences across filter banks are comparatively smaller in Figure 11a, consistent with Table 3 where mean Precision remains above 0.90 for all banks; nevertheless, best-case Precision occurs under inverse-Mel at 0.9948, while the lowest Precision floor appears under Linear Scale at 0.7888.

Importantly, these results should not be interpreted as replacing the main proposed inverse-Mel configuration. The Inverse Mel Scale filter bank produced both the highest mean F1-score and the global peak F1-score, supporting its use in the main MD-Emph inverse-Mel configuration. At the same time, its wider dispersion indicates that the final performance also depends on the feature-processing method and learning algorithm. Therefore, the robustness analysis is interpreted primarily at the feature-generation level: Window-Variance-Emphasized Spectrum Data provides a more stable performance envelope than Mean-Difference-Emphasized Spectrum Data across the evaluated configurations, whereas Inverse Mel remains the filter-bank scale used in the best-performing proposed setting.

Table 3. Results by filter bank type.

Filter Bank	Precision (Mean $\pm$ Std)	Recall (Mean $\pm$ Std)	F1-Score (Mean $\pm$ Std)
Mel Scale	0.9039 ± 0.0441	0.8316 ± 0.1295	0.8662 ± 0.0845
Inverse Mel Scale	0.9275 ± 0.0516	0.8843 ± 0.1681	0.9054 ± 0.1215
Gammatone Scale	0.9085 ± 0.0462	0.8787 ± 0.1774	0.8934 ± 0.1217
Linear Scale	0.9112 ± 0.0477	0.852 ± 0.1574	0.8806 ± 0.1049

5.2.3. Performance Comparison Among Different Machine Learning Methods

As described in Section 5.1.2, the five algorithms in Table 4 represent complementary anomaly-detection paradigms applied to the same BLE fingerprinting pipeline: reconstruction-based deep learning for AE-DNN and AE-1D CNN, boundary-based one-class learning for OCSVM, isolation-based anomaly scoring for IF, and density-based outlier detection for LOF. Thus, Table 4 should be interpreted as an algorithm-level comparison under a shared feature extraction and authentication setting, rather than as a comparison of unrelated processing pipelines.

The model-level trends in Figure 11a–c are strongly supported by Table 4 and by the min–max statistics computed across all configurations. Autoencoder-DNN repeatedly achieves the strongest outcomes in Figure 11b,c, particularly under emphasis-based feature processing, and Table 4 confirms that Autoencoder-DNN attains the highest mean F1-score, 0.9069 ± 0.0883 , as well as the highest mean Recall, 0.9151 ± 0.1000 . The full-table extremes further quantify this dominance: Autoencoder-DNN achieves the global maximum F1-score of 0.9927 and reaches maximum Recall 0.9912, matching the near-ceiling results visible in Figure 11b,c.

Classical detectors show larger variability, largely driven by Recall degradation in Figure 11b. Isolation Forest has the lowest mean F1-score, 0.7477 ± 0.0965 , and its Recall ranges from 0.4568 to 0.8478, explaining the weaker outcomes seen in Figure 11b and the corresponding F1-score limitations in Figure 11c. One-Class SVM often achieves high Precision in Figure 11a, consistent with mean Precision 0.9186 ± 0.0436 and maximum Precision 0.9750, but its Recall ranges widely from 0.4664 to 0.9406, which constrains F1-score stability. Local Outlier Factor shows a similar pattern: mean Precision is 0.9036 ± 0.0310 and maximum Precision 0.9605, yet Recall ranges from 0.4163 to 0.9638, producing a broad spread in Figure 11b,c. Autoencoder-1D CNN remains competitive overall but exhibits sensitivity under Raw settings; it shows the lowest Precision floor, 0.5578, and the lowest F1-score floor, 0.5572, consistent with the visibly degraded outcomes in Figure 11a,c for those cases.

Taken together, Figure 11a–c and Table 4 support selecting Autoencoder-DNN as the learning algorithm for the final configuration. Even when the absolute peak F1-score is achieved only under particular combinations, Autoencoder-DNN provides the most reliable overall behavior by sustaining high Recall while maintaining high Precision across diverse feature-processing and filter-bank settings.

Table 4. Results by algorithm type.

Algorithm	Precision (Mean $\pm$ Std)	Recall (Mean $\pm$ Std)	F1-Score (Mean $\pm$ Std)
IF	0.8658 ± 0.0564	0.6579 ± 0.1269	0.7477 ± 0.0965
OCSVM	0.9186 ± 0.0436	0.7881 ± 0.1689	0.8484 ± 0.1219
LOF	0.9036 ± 0.0310	0.7945 ± 0.1920	0.8455 ± 0.1237
AE-1D CNN	0.8790 ± 0.0939	0.8045 ± 0.0995	0.8401 ± 0.0888
AE-DNN	0.8988 ± 0.0791	0.9151 ± 0.1000	0.9069 ± 0.0883

In terms of authentication-oriented error characteristics, the method achieved a False Acceptance Rate (FAR) of 0.0444 and a False Rejection Rate (FRR) of 0.0101. The low FRR indicates that legitimate transmitters are accepted reliably, which is desirable for usability in RF fingerprint-based authentication. At the same time, the FAR remains a key metric to further improve because reducing unauthorized acceptance is critical for strengthening security guarantees. Overall, these results validate that the proposed framework effectively learns complex BLE signal patterns and distinguishes legitimate and anomalous signals with high accuracy, while also providing a practically robust configuration suitable for BLE-based IoT applications.

5.2.4. Threshold Generalization Analysis

To evaluate whether the authentication threshold can be selected without using abnormal calibration labels, we compared the labeled F1-optimized threshold with three normal-only statistical thresholds. In the normal-only settings, abnormal samples and abnormal labels were excluded from threshold estimation. For all compared methods, anomaly scores were oriented so that larger values indicate more abnormal behavior. Thresholds were estimated on the calibration split and evaluated on the held-out test split. The median-based rule uses the median absolute deviation (MAD). Table 5 summarizes the macro-averaged results on a randomly selected and fixed subset of the BLE evaluation feature set used in the main experiments, covering five algorithms and five feature or filter-bank aliases under two evaluation variants. The labeled F1-optimized threshold achieved the highest mean F1-score of 0.9505. Among the normal-only alternatives, the 99th-percentile threshold achieved a mean F1-score of 0.9299, while the mean-plus-three-standard-deviation threshold showed a similar F1-score of 0.9280. These results indicate that fully normal-only thresholding can

reduce dependence on labeled abnormal calibration data, but the security-related error profile differs across thresholding rules. In particular, the 99th-percentile and mean-plus-three-standard-deviation rules retain relatively high F1-scores but increase FAR compared with labeled calibration, whereas the median-plus-three-scaled-MAD rule lowers FAR at the cost of higher FRR and lower F1-score. Therefore, normal-only thresholding is feasible for unlabeled deployment settings, but selecting a threshold that generalizes well while controlling unauthorized acceptance remains a security-usability trade-off and requires further improvement in future work.

Table 5. Threshold generalization summary on a randomly selected and fixed subset of the BLE evaluation feature set.

Threshold Method	F1	Accuracy	FAR	FRR
Labeled F1-opt	0.9505	0.9374	0.1794	0.0292
P99 normal-only	0.9299	0.9043	0.3188	0.0153
Mean plus 3 standard deviations	0.9280	0.9018	0.3163	0.0166
Median plus 3 scaled MAD	0.9135	0.8878	0.1495	0.0952

5.2.5. Signal-Level Authentication Evaluation

Because the practical authentication decision is made for a BLE signal rather than for an isolated frame, we aggregated frame-level decisions into signal-level decisions. For each signal, frame-level anomaly scores were first converted into normal or abnormal frame decisions using the selected threshold. A signal was then accepted as normal when at least n frames were classified as normal; otherwise, it was rejected as abnormal. For threshold-independent signal-level ROC analysis, the signal anomaly score was computed as the abnormal-frame ratio, defined as the number of abnormal frames divided by the total number of frames in the signal. Table 6 compares the three aggregation-threshold rules described above: majority voting, labeled F1-optimized calibration, and normal-only 99% normal-signal acceptance. ROC-AUC and EER are identical across the three aggregation-threshold rules because they are computed from the threshold-independent abnormal-frame ratio, whereas F1-score, Accuracy, FAR, and FRR are computed after applying each fixed aggregation threshold. The majority rule achieved the strongest mean signal-level F1-score of 0.9685, with a mean ROC-AUC of 0.9521. These results confirm that signal-level aggregation preserves, and in this subset improves, the frame-level authentication behavior while better matching the intended authentication unit.

Table 6. Signal-level authentication summary using frame-to-signal aggregation.

Rule for n	F1	Accuracy	FAR	FRR	ROC-AUC	EER
Majority rule	0.9685	0.9577	0.1388	0.0139	0.9521	0.0601
Labeled F1-opt n	0.9538	0.9392	0.2006	0.0092	0.9521	0.0601
Normal-only 99% acceptance n	0.9386	0.9272	0.1866	0.0609	0.9521	0.0601

5.2.6. Sensitivity to the Pre-Emphasis Coefficient

We also examined the sensitivity of the authentication pipeline to the pre-emphasis coefficient α . The coefficient was swept from 0.90 to 0.99 in increments of 0.01, and the features were regenerated for each coefficient value before model training and evaluation. Because this experiment changes a preprocessing coefficient before feature generation, it is interpreted as a coefficient-sensitivity analysis rather than as a replacement for the main feature-comparison results. Table 7 reports the results for all ten tested coefficient values. Across the full sweep, the frame-level F1-score remained between 0.9333 and 0.9339, and the signal-level F1-score

remained between 0.9757 and 0.9766. The frame-level AUC also stayed within 0.9436–0.9457, while the signal-level AUC remained within 0.9863–0.9881. These results indicate that the final authentication outcome maintains high performance and is not highly sensitive to small changes in α within the tested range. Therefore, the use of $\alpha = 0.97$ in the main experiments is consistent with common pre-emphasis practice and is supported by the observed stability of authentication performance across the tested coefficient range.

Table 7. Pre-emphasis coefficient sensitivity summary for all tested alpha values.

Alpha	Frame F1	Frame Accuracy	Frame AUC	Signal F1	Signal Accuracy	Signal AUC
0.90	0.9335	0.8742	0.9455	0.9757	0.9354	0.9866
0.91	0.9338	0.8740	0.9453	0.9762	0.9367	0.9880
0.92	0.9339	0.8738	0.9447	0.9760	0.9351	0.9871
0.93	0.9338	0.8736	0.9457	0.9757	0.9357	0.9879
0.94	0.9333	0.8733	0.9438	0.9760	0.9362	0.9881
0.95	0.9335	0.8732	0.9441	0.9765	0.9370	0.9869
0.96	0.9334	0.8727	0.9436	0.9760	0.9353	0.9863
0.97	0.9336	0.8737	0.9456	0.9763	0.9371	0.9872
0.98	0.9335	0.8735	0.9444	0.9759	0.9358	0.9865
0.99	0.9336	0.8741	0.9455	0.9766	0.9371	0.9868

5.2.7. Comparison with Recent Deep-Learning-Based RF Fingerprinting Baselines

For comparison with recent deep-learning-based RF fingerprinting methods, we implemented architecture-adapted representative baselines inspired by channel-robust single-source domain generalization, open-world RFFI, and adaptive semantic augmentation studies [21–23]. These baselines were used to position the proposed framework against contemporary neural architectures rather than to reproduce the full original training objectives of the cited studies, which target different datasets and deployment assumptions. Unlike the classifier-ablation experiment on the proposed emphasized features, this comparison starts from raw time-domain BLE waveform recordings and does not use the proposed emphasized filter-bank data as the input to the related-work baselines. Each baseline was instead provided with an input representation matched to its model structure: a root-mean-square (RMS)-normalized raw waveform for SDG/TWC MACNN, a non-overlapping waveform-token sequence for OpenRFI/Roinformer, and a log-power short-time Fourier transform (STFT) representation followed by the discrete wavelet transform (DWT)-based ASA/TIFS spectral decomposition pipeline. Table 8 reports the best-performing configuration of the proposed framework from the main 65-configuration evaluation and compares it with the recent DL-RFFI baselines implemented from raw time-domain waveform recordings. Because the proposed row uses the emphasized spectral pipeline whereas the baseline rows use raw-waveform-derived representations tailored to each model architecture, the results should be interpreted as a comparison between the proposed end-to-end framework and architecture-adapted representative recent DL-RFFI baselines, with the input representation of each method explicitly reported.

Table 8. Recent DL-based RF fingerprinting baseline comparison.

Method	Input	Precision	Recall	F1	Accuracy	ROC-AUC
Proposed AE-DNN	MD-Emph inverse-Mel (main)	0.9948	0.9901	0.9927	0.9788	0.9820
SDG/TWC MACNN	Raw waveform	0.8581	0.9787	0.9138	0.8880	0.8818
OpenRFI/Roinformer	Raw waveform tokens	0.8102	0.8827	0.8446	0.8048	0.8695
ASA/TIFS	Raw STFT/DWT representation	0.8770	0.9227	0.8979	0.8736	0.9392

5.3. Per-Device Authentication Performance Analysis

Figure 12 illustrates the authentication performance for a representative enrolled device under the best-performing configuration. Figure 12a shows the Receiver Operating Characteristic curve, where an area under the curve close to 1 indicates strong discriminability between legitimate and anomalous signals. Figure 12b presents the Precision–Recall curve, depicting the trade-off between Precision and Recall across varying decision thresholds. Figure 12c shows the Confusion Matrix at the selected optimal threshold. The per-device results confirm that the proposed framework achieves reliable separation between normal and anomalous signals at the individual device level.

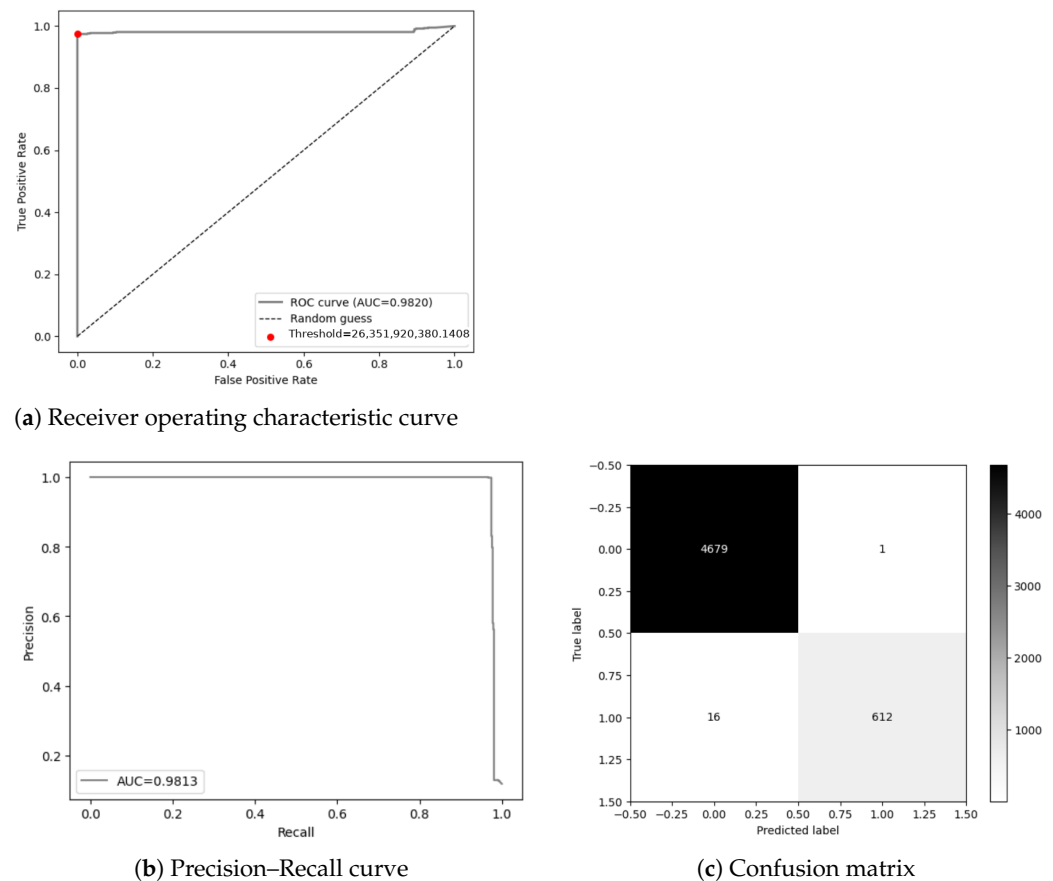


Figure 12. Performance examples for a representative device.

Figure 13 shows the macro-averaged ROC curve computed on the unseen test dataset, where the anomaly detection performance is averaged across all 40 enrolled devices with equal weighting. The x-axis represents the False Positive Rate (FPR) and the y-axis represents the True Positive Rate (TPR); the black dashed line indicates the chance-level baseline of a random classifier. The gray solid line is the mean ROC curve averaged over all 40 devices, summarizing the overall signal-separation performance of the proposed framework. The light gray shaded region represents ± 1 standard deviation of the TPR at each FPR operating point, visualizing inter-device performance variability across enrolled transmitters. The narrow width of the shaded region indicates that most devices achieve consistently similar detection performance. The macro-averaged AUC of approximately 0.94 demonstrates substantially higher separation capability compared to random classification. Notably, high TPR is maintained even at low FPR operating points, confirming that anomalous signals can be effectively detected while minimizing false acceptances. These results suggest that the proposed framework provides stable and generalizable anomaly detection performance despite hardware-dependent differences among transmitters.

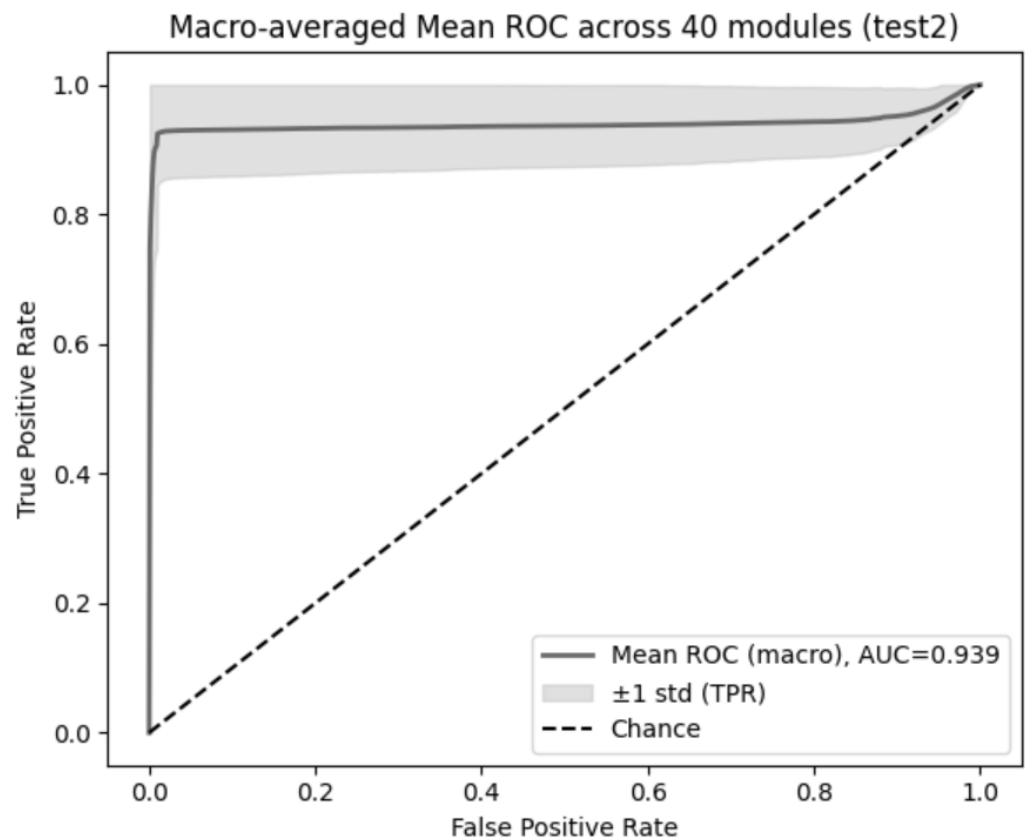


Figure 13. Performance on the unseen test dataset: Macro-averaged ROC.

To further characterize the error profile and threshold-selection behavior of the proposed authentication framework, Figure 14 presents the Detection Error Tradeoff (DET) curve together with the FAR/FRR-versus-threshold analysis evaluated on the unseen test dataset. Whereas the ROC curve primarily highlights overall discriminative performance, the DET curve focuses explicitly on operational error behavior, making it particularly suitable for security-oriented evaluation in which both unauthorized device acceptance and legitimate device rejection must be minimized.

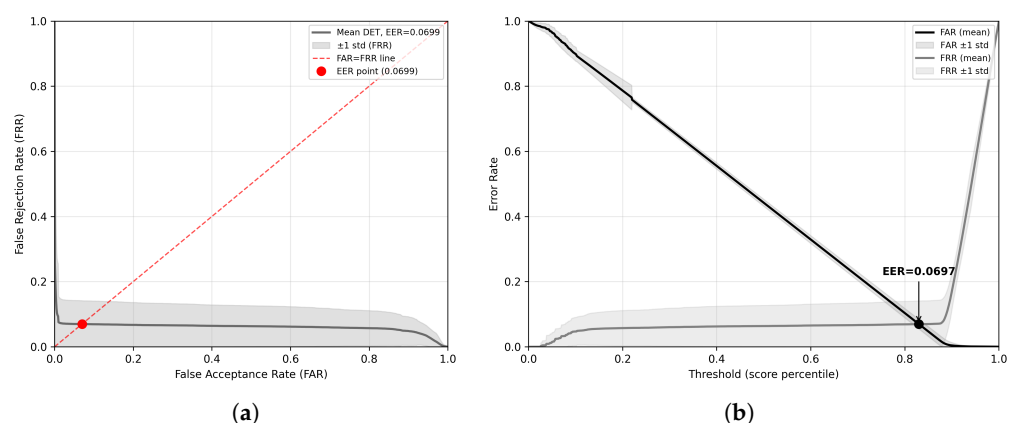


Figure 14. Error tradeoff and threshold analysis for 40 enrolled devices. (a) FRR-FAR tradeoff curve with the EER operating point. (b) FAR and FRR versus the normalized decision threshold, showing the EER point.

Figure 14a shows the mean FRR-versus-FAR tradeoff curve (solid black line), obtained by interpolating the device-specific curves onto a common FAR grid and averaging them across all 40 enrolled transmitters. The gray shaded region represents ± 1 standard deviation.

tion, reflecting inter-device variability. The concentration of the curve near the lower-left region indicates favorable authentication performance, with both false acceptance and false rejection remaining low around the principal operating region. The red dashed diagonal line denotes the $FAR = FRR$ condition, and its intersection with the mean curve defines the Equal Error Rate (EER). At this operating point, the proposed framework achieves an EER of 0.0699, as marked by the red dot in Figure 14a.

Figure 14b presents FAR (solid black line) and FRR (solid gray line) as functions of the normalized decision threshold expressed as a score percentile from 0.0 to 1.0. Each curve represents the mean over all enrolled devices, and the shaded regions indicate ± 1 standard deviation. As the threshold increases, FAR decreases monotonically whereas FRR increases monotonically, illustrating the inherent tradeoff between security and usability in threshold-based authentication. The two curves intersect at a threshold percentile of approximately 0.83–0.85, where the framework attains an EER of 0.0697.

Several practical implications can be drawn from Figure 14. First, the framework exhibits a clear operating region around the EER threshold in which both FAR and FRR remain below 0.1, indicating a practically useful balance between security and usability. Second, FRR remains low over most of the threshold range and increases sharply only in the high-threshold regime, suggesting that legitimate devices are accepted reliably under typical operating conditions. Third, the relatively compact variance bands around the operationally relevant region, particularly near the EER threshold, indicate that the observed error behavior is reasonably consistent across individual devices. These results suggest that the operating threshold can be tuned according to application requirements, allowing the system to favor stricter security or higher usability while maintaining stable authentication performance.

6. Conclusions and Future Work

We proposed a BLE signal fingerprinting framework for IoT device authentication that leverages emphasized-spectral features to enhance discriminability. By selectively identifying frequency regions exhibiting pronounced inter-device variability and applying device-specific filter banks with higher resolution in those regions, the proposed framework generates emphasized-spectral data that captures subtle hardware-dependent characteristics of BLE transmitters. Combined with an Autoencoder-DNN trained exclusively on normal device signals, this emphasis-based representation enables effective anomaly detection without using labeled attack data for model training. Threshold calibration can then be performed either with labeled calibration data or with normal-only statistical criteria, depending on deployment constraints.

Extensive experiments across 65 configurations confirmed that emphasis-based processing consistently outperforms both raw spectrum and uniformly filtered baselines. The best-performing and main proposed configuration, combining Mean-Difference-Emphasized Spectrum Data, Autoencoder-DNN, and the Inverse Mel Scale filter bank, achieved a Precision of 0.9948, Recall of 0.9901, and F1-score of 0.9927. The aggregate analysis further indicates that Window-Variance-Emphasized Spectrum Data can provide a more stable feature-generation alternative across varying configurations, suggesting that variability-based emphasis may be useful when robustness across feature-generation conditions is prioritized. At the aggregate level, the macro-averaged ROC AUC of approximately 0.94 across all 40 enrolled devices further confirms the framework's generalizable detection capability.

Furthermore, the FRR-versus-FAR tradeoff analysis and the FAR/FRR threshold analysis yielded EER values of 0.0699 and 0.0697, respectively, confirming that the proposed framework maintains balanced false-acceptance and false-rejection performance at the

equal-error operating point. The existence of a practically useful operating region around the EER threshold, in which both error rates remain below 0.1, further supports the framework's operational flexibility and robustness for BLE-based IoT authentication.

The threshold generalization analysis, signal-level evaluation, pre-emphasis sensitivity analysis, and comparison with recent DL-RFFI baselines clarify the deployment implications of the proposed framework. The threshold comparison shows that normal-only statistical thresholds can be used when abnormal calibration labels are unavailable, although with some loss relative to labeled F1-based calibration. The signal-level evaluation aligns the metric with the practical authentication unit by aggregating frame decisions back into BLE-signal decisions. The pre-emphasis sweep indicates that the selected coefficient is not a fragile operating point within the tested range, and the comparison with recent DL-RFFI baselines, conducted from raw time-domain BLE waveform recordings rather than from the proposed emphasized features, provides additional context for assessing the proposed emphasized-spectral pipeline relative to contemporary neural architectures. From a lightweight BLE deployment perspective, the computationally heavier emphasized-region identification and filter-bank construction can be performed during enrollment or model preparation. Online authentication then uses a fixed preprocessing pipeline, a predefined filter bank, and a trained anomaly detector, making the runtime procedure more compatible with resource-constrained IoT authentication than repeatedly redesigning the feature extractor at every authentication attempt.

In future work, we will focus on evaluating the framework under diverse environmental conditions, including varying distances, multipath interference, and different receiver hardware configurations. Adaptive threshold mechanisms that can compensate for temporal drift and channel variation will be investigated to improve deployment robustness. Additionally, extending the framework to open-set authentication scenarios with a larger device population and incorporating adversarial robustness testing against physical-layer replay and spoofing attacks are important directions for strengthening practical security guarantees.

Author Contributions: Conceptualization, H.P. and T.K.; methodology, H.P.; software, H.P.; validation, H.P. and G.C.; formal analysis, H.P.; investigation, H.P.; data curation, H.P.; writing—original draft preparation, H.P.; writing—review and editing, H.P., G.C. and T.K.; visualization, H.P.; supervision, T.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (IITP-2026-RS-2022-00164800); partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2025-25411243); partly supported by the 2025 Industrial Technology Alchemist Project funded by the Ministry of Trade, Industry & Energy (MOTIE) (RS-2025-02317769); and partly supported by a Korea University Grant.

Data Availability Statement: The BLE signal dataset used in this study is publicly available on Figshare at <https://doi.org/10.6084/m9.figshare.32030973>. The dataset contains per-signal power spectrum frames in NumPy format (.npy), derived from BLE baseband signals collected from 40 transmitter modules (500 signals per module). Each file stores spectral frames obtained after pre-emphasis, framing, windowing, and FFT processing, prior to any filter bank transformation or frequency-domain emphasis described in this paper.

Acknowledgments: This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (IITP-2026-RS-2022-00164800); partly supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2025-25411243); partly supported by the 2025 Industrial Technology

Alchemist Project funded by the Ministry of Trade, Industry & Energy (MOTIE) (RS-2025-02317769); and partly supported by a Korea University Grant.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

BLE	Bluetooth Low Energy	MSE	Mean Squared Error
IoT	Internet of Things	DNN	Deep Neural Network
RF	Radio Frequency	CNN	Convolutional Neural Network
SEI	Specific Emitter Identification	AE	Autoencoder
RFFI	Radio Frequency Fingerprint Identification	OCSVM	One-Class Support Vector Machine
DL	Deep Learning	IF	Isolation Forest
ISM	Industrial, Scientific, and Medical	LOF	Local Outlier Factor
MitM	Man-in-the-Middle	FAR	False Acceptance Rate
PSD	Power Spectral Density	FRR	False Rejection Rate
UUID	Universally Unique Identifier	FPR	False Positive Rate
PEPS	Passive Entry Passive Start	TPR	True Positive Rate
GFSK	Gaussian Frequency-Shift Keying	ROC	Receiver Operating Characteristic
OFDM	Orthogonal Frequency-Division Multiplexing	AUC	Area Under the Curve
RMS	Root Mean Square	EER	Equal Error Rate
UFB	Uniform Filter Bank	DET	Detection Error Tradeoff
FFT	Fast Fourier Transform	MAD	Median Absolute Deviation
STFT	Short-Time Fourier Transform	MD	Mean Difference
DFT	Discrete Fourier Transform	WV	Window Variance
DWT	Discrete Wavelet Transform	L1	L1 Regularization (Lasso)
FIR	Finite Impulse Response	L2	L2 Regularization (Ridge)
ELU	Exponential Linear Unit	Adam	Adaptive Moment Estimation
ReLU	Rectified Linear Unit		

Appendix A

Table A1. Overall experimental results for all feature and algorithm configurations.

Data Type	Algorithm	Filter Bank	Precision	Recall	F1-Score
Uniform FB Spectrum	IF	Mel Scale	0.9315	0.7291	0.8179
		Inverse Mel Scale	0.8538	0.5272	0.6519
		Gammatone Scale	0.8711	0.5062	0.6403
		Linear Scale	0.8746	0.4568	0.6002
	OCSVM	Mel Scale	0.8795	0.6926	0.7749
		Linear Scale	0.8973	0.5703	0.6974
		Inverse Mel Scale	0.8460	0.4664	0.6013
		Gammatone Scale	0.8567	0.5260	0.6518
	LOF	Mel Scale	0.8654	0.6119	0.7169
		Inverse Mel Scale	0.8884	0.5023	0.6418
		Linear Scale	0.9394	0.5864	0.7220
		Gammatone Scale	0.8957	0.4163	0.5684
	AE-1D CNN	Linear Scale	0.8690	0.9067	0.8875
		Mel Scale	0.9034	0.8702	0.8865
		Gammatone Scale	0.8666	0.8817	0.8741
		Inverse Mel Scale	0.8824	0.8579	0.8700
	AE-DNN	Mel Scale	0.8775	0.9090	0.8930
		Inverse Mel Scale	0.9091	0.8449	0.8758
		Gammatone Scale	0.8346	0.9028	0.8673
		Linear Scale	0.8316	0.9001	0.8645

Table A1. Cont.

Data Type	Algorithm	Filter Bank	Precision	Recall	F1-Score
MD-Emph. Spectrum	IF	Mel Scale	0.7998	0.6525	0.7187
		Inverse Mel Scale	0.8068	0.6963	0.7475
		Linear Scale	0.7888	0.6555	0.7160
	OCSVM	Gammatone Scale	0.8095	0.6645	0.7299
		Mel Scale	0.9750	0.8787	0.9243
		Inverse Mel Scale	0.9713	0.8957	0.9320
	LOF	Linear Scale	0.9669	0.8832	0.9232
		Gammatone Scale	0.9029	0.8822	0.8924
		Mel Scale	0.8902	0.9604	0.9239
	AE-1D CNN	Inverse Mel Scale	0.8887	0.9635	0.9246
		Linear Scale	0.8887	0.9638	0.9247
		Gammatone Scale	0.8768	0.9601	0.9166
	AE-DNN	Gammatone Scale	0.9022	0.8551	0.8780
		Inverse Mel Scale	0.9378	0.8700	0.9026
		Linear Scale	0.8817	0.8108	0.8448
		Mel Scale	0.8571	0.7907	0.8225
		Gammatone Scale	0.9539	0.9829	0.9680
		Inverse Mel Scale	0.9948	0.9901	0.9927
		Linear Scale	0.9485	0.9912	0.9694
		Mel Scale	0.9434	0.9828	0.9624
WV-Emph. Spectrum	IF	Mel Scale	0.9111	0.6163	0.7352
		Inverse Mel Scale	0.9336	0.8478	0.8887
		Linear Scale	0.9406	0.8396	0.8872
	OCSVM	Gammatone Scale	0.9519	0.8237	0.8832
		Mel Scale	0.9213	0.9152	0.9182
		Inverse Mel Scale	0.9646	0.9406	0.9524
	LOF	Linear Scale	0.9251	0.8577	0.8901
		Gammatone Scale	0.9411	0.9184	0.9296
		Mel Scale	0.8896	0.8914	0.8905
	AE-1D CNN	Inverse Mel Scale	0.9465	0.8677	0.9054
		Linear Scale	0.9555	0.8877	0.9204
		Gammatone Scale	0.9605	0.8702	0.9131
	AE-DNN	Mel Scale	0.9297	0.6930	0.7941
		Inverse Mel Scale	0.9471	0.8304	0.8849
		Linear Scale	0.8910	0.7051	0.7872
		Gammatone Scale	0.8755	0.9239	0.8990
		Mel Scale	0.9719	0.9753	0.9740
		Inverse Mel Scale	0.9628	0.9713	0.9670
		Linear Scale	0.9350	0.9458	0.9404
		Gammatone Scale	0.9610	0.9564	0.9578
Raw Spectrum Data	IF	-	0.8429	0.5026	0.6297
	OCSVM	-	0.8690	0.5689	0.6876
	LOF	-	0.8895	0.6012	0.7175
	AE-1D CNN	-	0.5578	0.5565	0.5572
	AE-DNN	-	0.6934	0.6067	0.6472

References

1. Hassan, R.; Qamar, F.; Hasan, M.; Mohd Aman, A.; Ahmed, A. Internet of Things and Its Applications: A Comprehensive Survey. *Symmetry* **2020**, *12*, 1674. [\[CrossRef\]](#)
2. Duran, J.; Cuesta, E.; Martinez Quintero, J. Replay Attacks and Sniffing in Bluetooth Low Energy Communications with Mobile Phone. *Bull. Electr. Eng. Inform.* **2025**, *14*, 3969–3984. [\[CrossRef\]](#)
3. Zhang, Y.; Weng, J.; Dey, R.; Jin, Y.; Lin, Z.; Fu, X. Breaking Secure Pairing of Bluetooth Low Energy Using Downgrade Attacks. In *Proceedings of the 29th USENIX Security Symposium (USENIX Security 20)*, Boston, MA, USA, 12–14 August 2020; USENIX Association: Berkeley, CA, USA, 2020; pp. 37–54.
4. Wu, J.; Traynor, P.; Xu, D.; Tian, D.; Bianchi, A. Finding Traceability Attacks in the Bluetooth Low Energy Specification and Its Implementations. In *Proceedings of the 33rd USENIX Security Symposium (USENIX Security 24)*, Philadelphia, PA, USA, 14–16 August 2024; USENIX Association: Berkeley, CA, USA, 2024; pp. 4499–4516.
5. Perri, M.; Cuomo, F.; Locatelli, P. BLENDER: Bluetooth Low Energy Discovery and Fingerprinting in IoT. In *Proceedings of the 20th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Pafos, Cyprus, 1–3 June 2022; IEEE: New York, NY, USA, 2022; pp. 182–189. [\[CrossRef\]](#)

6. Locatelli, P.; Perri, M.; Jimenez Gutierrez, D.; Lacava, A.; Cuomo, F. Device Discovery and Tracing in the Bluetooth Low Energy Domain. *Comput. Commun.* **2023**, *202*, 42–56. [\[CrossRef\]](#)
7. Zuo, C.; Wen, H.; Lin, Z.; Zhang, Y. Automatic Fingerprinting of Vulnerable BLE IoT Devices with Static UUIDs from Mobile Apps. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS 2019), London, UK, 11–15 November 2019*; ACM: New York, NY, USA, 2019; pp. 1469–1483. [\[CrossRef\]](#)
8. Yaseen, M.; Iqbal, W.; Rashid, I.; Abbas, H.; Mohsin, M.; Saleem, K.; Bangash, Y. MARC: A Novel Framework for Detecting MITM Attacks in eHealthcare BLE Systems. *J. Med. Syst.* **2019**, *43*, 324. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Yurdagul, M.; Sencar, H. BLEKeeper: Response Time Behavior Based Man-In-The-Middle Attack Detection. In *Proceedings of the IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 27 May 2021*; IEEE: New York, NY, USA, 2021; pp. 214–220. [\[CrossRef\]](#)
10. Lahmadi, A.; Duque, A.; Heraief, N.; Francq, J. MitM Attack Detection in BLE Networks Using Reconstruction and Classification Machine Learning Techniques. In *Proceedings of the ECML PKDD 2020 Workshops, Ghent, Belgium, 14–18 September 2020*; Communications in Computer and Information Science; Springer International Publishing: Cham, Switzerland, 2020; Volume 1323, pp. 149–164. [\[CrossRef\]](#)
11. Galtier, F.; Cayre, R.; Auriol, G.; Kâaniche, M.; Nicomette, V. A PSD-Based Fingerprinting Approach to Detect IoT Device Spoofing. In *Proceedings of the 25th Pacific Rim International Symposium on Dependable Computing (PRDC), Perth, WA, Australia, 1–4 December 2020*; IEEE: New York, NY, USA, 2020; pp. 40–49. [\[CrossRef\]](#)
12. Wu, J.; Nan, Y.; Kumar, V.; Payer, M.; Xu, D. BlueShield: Detecting Spoofing Attacks in Bluetooth Low Energy Networks. In *Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020), San Sebastian, Spain, 14–16 October 2020*; USENIX Association: Berkeley, CA, USA, 2020; pp. 397–411.
13. Abu Al-Haija, Q.; Alsulami, A. Detection of Fake Replay Attack Signals on Remote Keyless Controlled Vehicles Using Pre-Trained Deep Neural Network. *Electronics* **2022**, *11*, 3376. [\[CrossRef\]](#)
14. Bonavolontà, F.; Liccardo, A.; Schiano Lo Moriello, R.; Caputo, E.; de Alteriis, G.; Palladino, A.; Vitolo, G. An Improved Method Based on Bluetooth Low-Energy Fingerprinting for the Implementation of PEPS System. *Sensors* **2022**, *22*, 9615. [\[CrossRef\]](#) [\[PubMed\]](#)
15. Liu, M.; Han, X.; Liu, N.; Peng, L. Bidirectional IoT Device Identification Based on Radio Frequency Fingerprint Reciprocity. In *Proceedings of the IEEE International Conference on Communications (ICC 2021), Montreal, QC, Canada, 14–23 June 2021*; IEEE: New York, NY, USA, 2021; pp. 1–6. [\[CrossRef\]](#)
16. Zhang, J.; Ardizzon, F.; Piana, M.; Shen, G.; Tomasin, S. Physical Layer-Based Device Fingerprinting for Wireless Security: From Theory to Practice. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 5296–5325. [\[CrossRef\]](#)
17. Stoian, G.A.; Voigt, T.; Rohner, C. Augmenting BLE Fingerprinting Using Instantaneous Frequency. In *Proceedings of the 18th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec 2025), Arlington, VA, USA, 30 June–3 July 2025*; ACM: New York, NY, USA, 2025; pp. 274–279. [\[CrossRef\]](#)
18. Zhang, J.; Li, X.; Li, J.; Dai, Q.; Ling, Z.; Yang, M. Bluetooth Low Energy Device Identification Based on Link Layer Broadcast Packet Fingerprinting. *Tsinghua Sci. Technol.* **2023**, *28*, 862–872. [\[CrossRef\]](#)
19. Sun, X.; Dang, F. FingerBLE: A Device Fingerprint Identification Scheme for BLE Devices. In *Proceedings of the IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS), Ocean Flower Island, Danzhou, China, 17–21 December 2023*; IEEE: New York, NY, USA, 2023; pp. 907–912. [\[CrossRef\]](#)
20. Shen, G.; Zhang, J.; Wang, X.; Mao, S. Federated Radio Frequency Fingerprint Identification Powered by Unsupervised Contrastive Learning. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 9204–9215. [\[CrossRef\]](#)
21. Wang, Y.; Ohtsuki, T.; Sun, Z.; Niyato, D.; Wang, X.; Gui, G. Avoiding Shortcuts: Enhancing Channel-Robust Specific Emitter Identification via Single-Source Domain Generalization. *IEEE Trans. Wirel. Commun.* **2025**, *24*, 3163–3176. [\[CrossRef\]](#)
22. Han, Z.; Xiao, J.; Zhao, Q.; Cui, Z.; Wang, Y.; Zhang, D.; Ding, W. Open-world Radio Frequency Fingerprint Identification via Augmented Semi-supervised Learning. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence (AAAI-25), Philadelphia, PA, USA, 25 February–4 March 2025*; AAAI Press: Washington, DC, USA, 2025; pp. 264–272. [\[CrossRef\]](#)
23. Cai, Z.; Wang, Y.; Gui, G.; Sha, J. Toward Robust Radio Frequency Fingerprint Identification via Adaptive Semantic Augmentation. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 1037–1048. [\[CrossRef\]](#)
24. Pallavi, S.; Narayanan, V. An Overview of Practical Attacks on BLE-Based IoT Devices and Their Security. In *Proceedings of the 5th International Conference on Advanced Computing & Communication Systems (ICACCS), Coimbatore, India, 15–16 March 2019*; IEEE: New York, NY, USA, 2019; pp. 694–698. [\[CrossRef\]](#)

25. Lacava, A.; Giacomini, E.; D’Alterio, F.; Cuomo, F. Intrusion Detection System for Bluetooth Mesh Networks: Data Gathering and Experimental Evaluations. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Kassel, Germany, 22–26 March 2021*; IEEE: New York, NY, USA, 2021; pp. 661–666. [[CrossRef](#)]
26. Gu, T.; Fang, Z.; Abhishek, A.; Fu, H.; Hu, P.; Mohapatra, P. IoTGaze: IoT Security Enforcement via Wireless Context Analysis. In *Proceedings of the IEEE INFOCOM Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020*; IEEE: New York, NY, USA, 2020; pp. 884–893. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.