

Article

Lightweight Multi-Attribute Physical Layer Authentication in LoRaWAN: A Comparative Study of Knowledge Distillation and Direct Training

Azita Pourghasem , Raimund Kirner , Athanasios Tsokanos, Iosif Mporas  and Alexios Mylonas 

Cybersecurity and Computing Systems Research Group, Department of Computer Science, University of Hertfordshire, Hatfield AL10 9AB, UK; a.mylonas@herts.ac.uk (A.M.)

* Correspondence: a.pourghasem@herts.ac.uk

Abstract

LoRaWAN has become a widely adopted communication technology for Internet of Things applications because of its long-range communication capability and low power consumption. However, resource-constrained LoRaWAN networks remain vulnerable to physical-layer attacks such as jamming and battery-depletion attacks. This paper investigates architectural compression and Knowledge Distillation (KD) for lightweight multi-attribute Physical Layer Authentication (PLA) in LoRaWAN networks. A higher-capacity Teacher model is used to train progressively smaller Student models, and three Student-training strategies—Direct training, Hard-KD, and Soft-KD—are systematically compared to determine whether Teacher supervision provides an advantage beyond architectural compression alone. The framework uses the Received Signal Strength Indicator (RSSI), altitude, and the battery level as input attributes and is evaluated offline using a device-aware semi-synthetic dataset derived from the publicly available Brno LoRaWAN dataset. The resulting dataset contains Normal Communication, Jamming Attack, and Battery Depletion Attack traffic. The experimental results show that Student (8) preserves a classification performance close to that of the Teacher across all three training strategies, achieving Macro F1-scores of $90.619 \pm 0.056\%$ for Direct training, $90.591 \pm 0.069\%$ for Hard-KD, and $90.626 \pm 0.083\%$ for Soft-KD, compared with 91.007% for the Teacher. Student (8) reduces the number of trainable parameters from 2435 to 59 (97.6%) and the serialised model size from 46.65 KB to 2.80 KB (94.0%), while exhibiting a substantially higher measured batch-processing throughput under the evaluated CPU-based implementation. A feature-ablation analysis further indicates that the RSSI and battery level provide the dominant discriminatory information, while the altitude provides a smaller but measurable contextual contribution. The comparable performance of Direct, Hard-KD, and Soft-KD demonstrates that KD does not consistently outperform direct training for the evaluated Student architectures; rather, architectural compression itself accounts for much of the retained performance. These findings demonstrate that compact Student architectures can substantially reduce the model complexity while preserving a strong classification performance within the evaluated multi-attribute PLA framework, supporting the further investigation of lightweight network-side security mechanisms for LoRaWAN deployments.

Keywords: LoRaWAN; Physical Layer Authentication (PLA); Knowledge Distillation (KD); Internet of Things (IoT); machine learning; lightweight artificial intelligence; jamming attack detection; battery-depletion attack; model compression



Academic Editors: Fatima Salahdine and Naima Kaabouch

Received: 3 July 2026

Revised: 22 August 2026

Accepted: 25 August 2026

Published: 28 August 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](https://creativecommons.org/licenses/by/4.0/)

Attribution (CC BY) license.

1. Introduction

The rapid expansion of the Internet of Things (IoT) has enabled billions of interconnected devices to support applications including environmental monitoring, smart agriculture, industrial automation, healthcare, and smart cities [1]. Many of these applications rely on Low-Power Wide-Area Network (LPWAN) technologies that provide long-range communication while maintaining low power consumption and low deployment costs. Among the available LPWAN technologies, LoRaWAN has become one of the most widely adopted communication protocols because of its scalability, long-range connectivity, and suitability for battery-powered devices operating in resource-constrained environments [2–4]. As LoRaWAN is increasingly adopted in critical infrastructures, ensuring secure, reliable, and scalable communication has become a fundamental research challenge.

Although LoRaWAN incorporates cryptographic mechanisms to provide device authentication and data confidentiality, these protections operate primarily at the network and application layers and, therefore, offer a limited defence against attacks targeting the physical layer [5–8]. Physical-layer attacks, including spoofing, jamming, replay, and battery-depletion attacks, can significantly degrade the communication reliability, reduce the network availability, and shorten the device lifetime. Moreover, the limited computational capability, memory, and energy resources associated with IoT environments restrict the use of computationally intensive security mechanisms. Consequently, recent research has advocated complementing conventional cryptographic approaches with lightweight physical-layer security mechanisms capable of detecting malicious wireless behaviour [6–10].

PLA provides an additional layer of defence by exploiting the characteristics of wireless transmissions and device behaviour that can assist in distinguishing legitimate from malicious communication. Unlike conventional authentication methods that rely solely on cryptographic credentials, PLA can incorporate physical-layer and device-related information into the authentication process. Recent surveys show that machine-learning and deep-learning techniques have substantially advanced PLA by enabling an intelligent analysis of complex communication patterns and multiple physical-layer attributes [7–9]. More recently, generative artificial intelligence has been investigated for physical-layer authentication, including fingerprint augmentation, reconstruction, denoising, and robustness enhancement [10].

Building on these developments, our previous work proposed a multi-attribute PLA framework for LoRaWAN that combines the RSSI, battery level, and altitude to detect spoofing attacks using machine-learning and deep-learning techniques [11]. This framework was subsequently extended to detect jamming and battery-depletion attacks using the same multi-attribute authentication strategy [12]. The results demonstrated the feasibility of combining communication and device-related attributes for LoRaWAN attack detection. However, those studies primarily focused on authentication and attack-detection performance rather than the systematic reduction of model complexity. Even when inference is performed at the network-side infrastructure rather than directly on an end device, reducing the model complexity can decrease the storage requirements and computational overhead while improving the processing efficiency.

The increasing adoption of lightweight machine learning and Edge AI has strengthened the interest in executing intelligent processing closer to where data are generated while reducing the communication and computational overhead. At the same time, Tiny Machine Learning (TinyML) has demonstrated the feasibility of executing compact machine-learning models on highly resource-constrained embedded platforms. Developments in lightweight machine learning, Edge AI, and TinyML have, therefore, highlighted the importance of reducing the model complexity through techniques such as compact architecture design, pruning, quantisation, and knowledge transfer [13–16]. Although the present study eval-

uates network-side rather than embedded-device inference, these developments provide broader motivation for investigating computationally efficient machine-learning models for large-scale IoT security.

Among the techniques proposed for model compression, Knowledge Distillation (KD) transfers predictive information from a higher-capacity Teacher model to a compact Student model. The classical formulation introduced by Hinton et al. [17] uses softened output probabilities to convey information about the Teacher's relative confidence across classes. KD has subsequently been investigated extensively in areas including computer vision and object detection [18] and has developed into a broader family of model-compression and knowledge-transfer techniques [19]. Teacher–Student learning has also been investigated in resource-aware Edge-AI and federated-learning environments [20].

However, strong performance from a distilled Student does not necessarily demonstrate that KD itself is responsible for the retained performance. A compact architecture may already possess sufficient representational capacity to learn the classification task directly from the original ground-truth labels. Consequently, evaluating a distilled Student without an equivalent directly trained Student baseline does not establish whether any observed benefit results specifically from KD or from architectural compression itself. This distinction is particularly important when assessing lightweight models intended for computationally efficient security applications.

This work addresses that issue by investigating whether the previously developed multi-attribute LoRaWAN PLA model can be substantially compressed while preserving the classification performance within the multi-attribute PLA framework and whether KD provides measurable benefits beyond the direct training of the same compact architectures. A higher-capacity Multi-Layer Perceptron (MLP) is used as the Teacher, while progressively smaller Student MLPs containing 8, 4, and 2 hidden neurons are evaluated. For each Student architecture, three training strategies are compared: Direct training using the original ground-truth labels, Hard-KD using Teacher-generated class labels, and Soft-KD using softened Teacher output distributions together with ground-truth supervision. Student experiments are repeated across five random seeds to assess the performance stability under increasing architectural compression. In addition, feature-ablation and computational-efficiency analyses are conducted to examine attribute contribution and the trade-off between the predictive performance and model complexity.

The main contributions of this work are summarised as follows:

- A controlled comparison of Direct, Hard-KD, and Soft-KD training is conducted using identical compact Student architectures, enabling the effect of Teacher supervision to be distinguished from architectural compression alone. The results show that KD does not consistently outperform Direct training for the evaluated Student architectures, indicating that architectural compression itself accounts for much of the retained performance.
- Three progressively compressed Student architectures containing 8, 4, and 2 hidden neurons are evaluated across five random seeds to quantify the classification-performance retention, stability, and minority-class detection under increasing architectural compression.
- A feature-ablation analysis quantifies the individual and combined contributions of the RSSI, altitude, and battery level, showing that the RSSI and battery level provide the dominant discriminatory information while the altitude contributes a comparatively modest additional benefit.
- A computational-efficiency evaluation quantifies reductions in the trainable parameters and serialised model size together with the CPU-based inference latency and throughput, providing an offline assessment of the performance–complexity trade-off under the evaluated computational environment.

The remainder of this paper is organised as follows. Section 2 reviews the related work on LoRaWAN security, PLA, lightweight artificial intelligence, and KD. Section 3 presents the proposed lightweight multi-attribute PLA approach, dataset preparation, training strategies, and Teacher–Student architectures. Section 4 presents the experimental results, including the classification performance, feature-ablation analysis, and computational efficiency. Finally, Section 5 concludes the paper and discusses the principal findings and limitations.

2. Related Work

Recent research has substantially advanced LoRaWAN security through developments in physical-layer security, machine-learning-based authentication, and lightweight artificial intelligence. This section reviews research on the LoRaWAN security and physical-layer threats, PLA, and lightweight model optimisation, with particular emphasis on KD, and identifies the research gap addressed by the present study.

2.1. LoRaWAN Security and Physical-Layer Threats

LoRaWAN security has traditionally relied on cryptographic mechanisms implemented at the network and application layers to provide authentication, integrity, and confidentiality [5,6]. Although these mechanisms are essential for protecting network communication, they do not directly prevent all attacks affecting the radio channel or network availability. Furthermore, the resource constraints associated with IoT devices have motivated the development of lightweight authentication mechanisms that seek to provide effective security while limiting computational and communication overhead [21].

Physical-layer threats present an additional challenge because they can affect wireless communication independently of the validity of higher-layer cryptographic credentials. Jamming, for example, can disrupt legitimate transmissions by interfering with the radio channel, while other availability-oriented attacks can increase communication or energy consumption and reduce the device lifetime. Reviews of LoRaWAN physical-layer attacks have identified jamming and related threats as important security concerns and have highlighted the need for complementary countermeasures [22].

Physical-layer security has therefore been investigated as an additional defence mechanism for wireless and IoT networks. Rather than replacing conventional cryptographic protection, physical-layer techniques exploit the characteristics of wireless signals, channels, or device behaviour to provide additional information for identifying anomalous or unauthorised communication [7,8]. This complementary role is particularly relevant to LoRaWAN, where communication reliability can be affected by both malicious interference and variations in the wireless environment.

2.2. Physical Layer Authentication and Machine-Learning-Based Approaches

PLA provides an additional authentication and security mechanism by exploiting physical-layer or device-related characteristics associated with wireless transmissions [7–9]. Depending on the application, PLA techniques may use attributes such as the RSSI, Channel State Information (CSI), carrier-frequency characteristics, or Radio Frequency Fingerprinting (RFF) to distinguish legitimate devices and communication behaviour from anomalous or malicious activity [7–9].

A major challenge for physical-layer authentication is that wireless characteristics can be affected by environmental and deployment conditions. Channel variation, receiver hardware, physical location, protocol configuration, and other operating conditions may influence learned signal representations and, consequently, affect model generalisation. The experimental research on LoRa device fingerprinting has demonstrated that changes

in network deployment conditions can substantially affect the performance of learned device fingerprints, illustrating the importance of robustness when applying physical-layer characteristics in practical environments [23].

Radio Frequency Fingerprinting represents a prominent branch of PLA in which distinctive transmitter characteristics are used to identify wireless devices. Traditional and deep-learning-based RFF techniques have demonstrated considerable potential for device identification, although challenges remain in relation to environmental variation, receiver dependence, dataset construction, scalability, and generalisation [24]. These challenges motivate the investigation of complementary attributes that are not dependent on a single radio-frequency characteristic.

Machine-learning and deep-learning methods have extended PLA beyond manually defined thresholds and statistical decision rules by learning discriminative patterns from the communication data. Recent surveys classify machine-learning-based PLA approaches according to tasks such as device identification and attack detection and highlight the increasing use of learning-based techniques for modelling complex physical-layer characteristics [8,9]. Generative artificial intelligence has also recently been investigated for physical-layer authentication, including applications involving fingerprint augmentation, reconstruction, denoising, and robustness enhancement [10].

Research has increasingly considered the use of multiple communication and device-related attributes rather than relying on a single physical-layer characteristic. Combining complementary attributes can provide a broader representation of communication behaviour and reduce the dependence on any individual signal characteristic [8,9]. Building on this direction, our previous work proposed a multi-attribute PLA framework for LoRaWAN incorporating the RSSI, battery level, and altitude for spoofing detection [11]. The framework was subsequently extended to jamming and battery-depletion attack detection using a semi-synthetic LoRaWAN dataset [12].

Broader security analyses of LoRaWAN have similarly demonstrated that the protocol is exposed to security risks across multiple layers and that effective protection requires the consideration of both protocol-level and wireless-channel threats [25]. These findings further motivate complementary security mechanisms capable of incorporating communication and device-related information without replacing the cryptographic protections already provided by LoRaWAN.

Our previous studies established the feasibility of the multi-attribute PLA approach, but their primary objective was authentication and attack-detection capability rather than the systematic compression of the classification model [11,12]. Consequently, the present study addresses a different question: whether the classification model can be substantially reduced in complexity while maintaining its predictive performance, and whether KD provides an advantage over directly training an equivalent compact architecture.

2.3. Lightweight AI and Knowledge Distillation for IoT

The increasing use of machine learning in IoT has motivated substantial research into efficient inference on platforms with limited memory, computational capability, and energy resources. Edge AI moves intelligent processing closer to the data source, while TinyML focuses particularly on executing compact machine-learning models on resource-constrained embedded platforms [13–16]. Research in these areas has highlighted the model complexity, memory requirements, execution efficiency, and hardware constraints as important considerations for practical machine-learning deployment.

A range of techniques can be used to reduce the machine-learning complexity, including compact architecture design, pruning, quantisation, and KD. KD differs from direct parameter-reduction techniques by transferring predictive information from a higher-

capacity Teacher model to a smaller Student model. In the classical formulation introduced by Hinton et al. [17], temperature-scaled output probabilities expose information about the Teacher's relative confidence across classes and can be used as additional supervision during Student training.

KD has subsequently been investigated extensively in computer-vision applications, including object detection [18], and has evolved into a broader family of model-compression and knowledge-transfer techniques [19]. Teacher–Student learning has also been incorporated into resource-aware Edge-AI and federated-learning frameworks, demonstrating its applicability to distributed and computationally constrained learning environments [20]. Collectively, these studies demonstrate the potential of Teacher–Student learning when a compact model is required to approximate the predictive behaviour of a higher-capacity model.

However, a strong performance from a compact Student does not by itself demonstrate that KD is responsible for that performance. A reduced architecture may already possess sufficient representational capacity to learn the underlying classification task directly from the original ground-truth labels. Consequently, evaluating a distilled Student without an equivalent directly trained Student baseline makes it difficult to distinguish the benefit of Teacher supervision from the effect of architectural compression itself.

Despite the extensive research on KD and lightweight machine learning more generally, comparatively little attention has been given to controlled KD evaluation within multi-attribute PLA for LoRaWAN. Our previous studies established multi-attribute PLA for spoofing, jamming, and battery-depletion detection [11,12], but did not systematically compare directly trained compact models with hard- and soft-distilled counterparts. The present work addresses this gap by comparing Direct, Hard-KD, and Soft-KD training across progressively compressed Student architectures while retaining the same underlying LoRaWAN PLA task. In addition, repeated-seed evaluation, feature-ablation analysis, and CPU-based computational benchmarking are used to assess the performance stability, attribute contribution, and performance–complexity trade-off without assuming operational embedded-hardware deployment.

3. Methodology

This section describes the methodology used to develop and evaluate the lightweight multi-attribute PLA approach for LoRaWAN networks. It presents the overall system architecture, dataset preparation, feature extraction, threat model, Teacher–Student learning process, and the evaluation methodology used to assess both classification performance and computational efficiency.

3.1. Proposed Framework and System Architecture

The proposed approach investigates architectural compression and alternative training strategies for developing lightweight multi-attribute PLA models for LoRaWAN networks. The approach operates at the LoRaWAN Network Server (NS), where communication records received from end devices are analysed using three attributes: RSSI, altitude, and battery level. These attributes are jointly used to distinguish among Normal Communication, Jamming Attack, and Battery Depletion Attack traffic.

In the context of this study, PLA is used as a complementary physical-layer-assisted security mechanism rather than as a conventional device-identity authentication mechanism. The classification task does not attempt to verify the cryptographic identity of an individual LoRaWAN end device. Instead, it evaluates whether the observed multi-attribute communication behaviour is consistent with Normal Communication or with the behavioural signatures associated with Jamming or Battery Depletion attacks. Accordingly, the term

PLA in this work refers to the use of physical-layer and device-related attributes within a complementary security framework for attack-state classification at the Network Server.

The threat model considered in this study focuses on availability-oriented attacks whose effects are observable through the attributes available at the Network Server. Jamming represents an adversarial condition in which interference disrupts legitimate LoRaWAN communication and is reflected through anomalous received-signal behaviour [12,22]. Battery Depletion represents an adversarial condition intended to accelerate device energy consumption and is reflected through abnormal battery-level behaviour [12]. The framework does not assume that these attacks compromise LoRaWAN cryptographic credentials or impersonate the identity of a legitimate end device. Rather, it uses the selected physical-layer and device-related attributes to identify communication behaviour that is inconsistent with the Normal class. In the evaluated semi-synthetic dataset, these attack effects are represented through packet-level metadata rather than through direct waveform-level measurements or operational over-the-air attack execution [12].

The overall framework follows a Teacher–Student architecture. A higher-capacity MLP, referred to as the Teacher model, is first trained offline using the ground-truth class labels. The knowledge captured by the trained Teacher is then used to guide the training of a substantially smaller Student model. The objective is to determine whether the reduced-capacity Student can retain classification performance close to that of the Teacher while requiring substantially fewer trainable parameters and less storage.

To determine whether any observed performance is attributable specifically to KD rather than simply to the capacity of the smaller neural network, three Student-training strategies are considered in the revised framework. First, Direct training trains the Student using the original ground-truth labels and, therefore, provides the principal non-KD baseline. Second, Hard-label KD trains the Student using the Teacher’s predicted class labels obtained through the argmax of its output probabilities. Third, Soft-label KD uses the Teacher’s class-probability distribution together with the ground-truth labels, allowing the Student to exploit information about the Teacher’s relative confidence across the three classes. Comparing these strategies enables the contribution of KD to be distinguished from the effect of model compression alone.

The Teacher is used only during offline training and is not required during Student inference. Following training, the lightweight Student model is evaluated as the inference model at the Network Server. This separation ensures that the computational requirements of the Teacher and the distillation procedure are confined to the offline training stage. The present study therefore evaluates the feasibility of reducing the complexity of the PLA classifier while retaining classification performance under the evaluated Network-Server-oriented inference setting.

Figure 1 illustrates the overall system architecture, including LoRaWAN end devices, gateways, the Network Server, the offline Teacher-training stage, and the lightweight Student model used for classification. The experimental evaluation is conducted offline using the semi-synthetic dataset described in Section 3.2; consequently, deployment on operational LoRaWAN hardware is not claimed in this study. Deployment on more constrained platforms, including gateway-level or end-device implementations, remains a potential direction for future investigation.

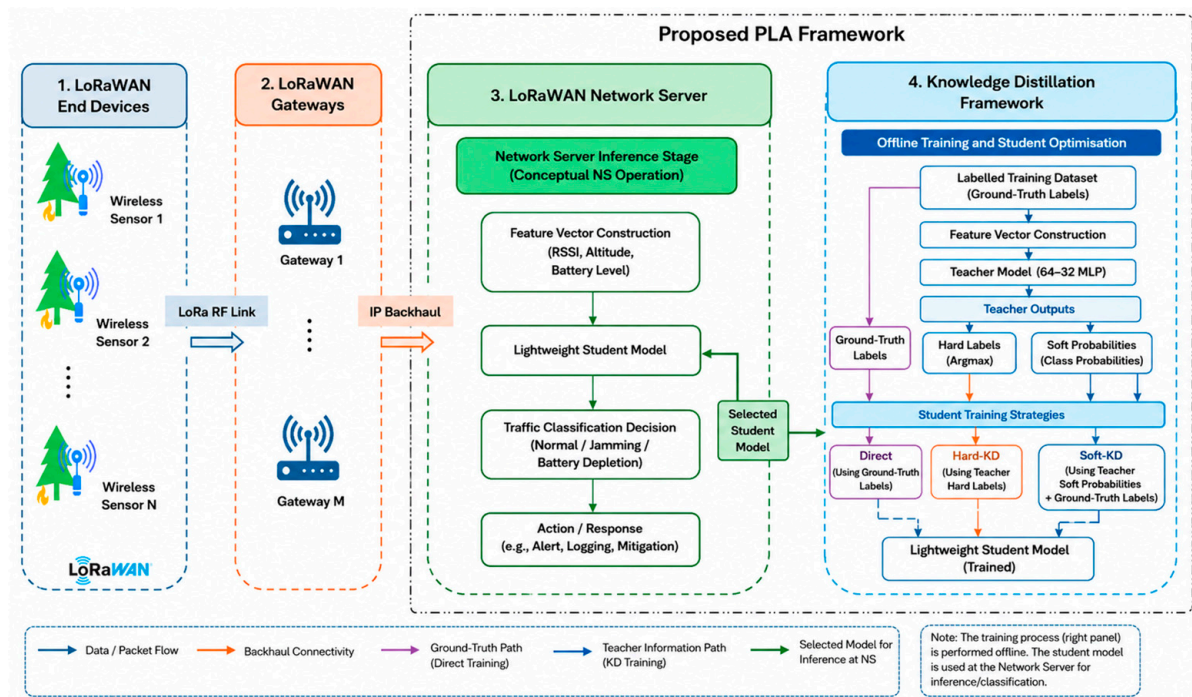


Figure 1. Proposed multi-attribute PLA framework for LoRaWAN, showing the conceptual Network Server inference stage and the offline Direct, Hard-KD, and Soft-KD Student-training strategies.

3.2. Dataset Preparation and Feature Engineering

This study uses the semi-synthetic LoRaWAN dataset developed and validated in our previous work on multi-attribute PLA for detecting jamming and battery-depletion attacks [12]. Rather than constructing a new dataset, the present study reuses the same dataset to investigate whether model compression and KD can reduce model complexity while preserving classification performance. The contribution of the present work therefore concerns lightweight model optimisation rather than the development of a new dataset or attack-generation mechanism. The complete processed semi-synthetic dataset used in the present study is publicly available through Zenodo [26].

The dataset was originally derived from the publicly available LoRaWAN Traffic Analysis Dataset [27], which contains communication records collected from operational LoRaWAN deployments, including measurements from Brno in the Czech Republic. The original dataset includes device identifiers and radio-related attributes such as RSSI, Signal-to-Noise Ratio (SNR), and Spreading Factor (SF). As described in our previous study [12], these communication traces were extended through a device-aware semi-synthetic augmentation process that preserved the original device structure, transmission order, and empirical radio characteristics while incorporating additional spatial, energy, and temporal attributes.

More specifically, augmentation was performed at the individual-device level while preserving transmission order. Each device was assigned an initial battery level, with approximately 65% of devices initialised within the 90–98% range and a minority initialised at lower levels to represent partially depleted nodes. The battery level was subsequently decreased incrementally over successive transmissions to represent normal energy consumption. Each device was also assigned a fixed installation altitude between 2.0 and 10.0 m, with small zero-mean Gaussian perturbations introduced to represent measurement variability [12].

Availability-related attack conditions were introduced using the behavioural, dataset-driven procedure described in [12]. Jamming behaviour was represented through anomalous RSSI patterns intended to emulate interference effects observable at the receiver, while

Battery Depletion behaviour was represented through accelerated energy consumption and abnormal battery-level reduction relative to normal depletion. These attack conditions were introduced at the packet-record level and were designed to represent attack effects observable from metadata available at the Network Server rather than direct waveform-level measurements or operational over-the-air attack execution. The augmentation and attack-labelling procedures were applied within the device-aware dataset construction process described in [12].

The resulting dataset contains 230,296 communication records from 1921 LoRaWAN devices and represents three traffic classes: Normal Communication, Jamming Attack, and Battery Depletion Attack. The overall class distribution comprises 208,255 Normal records (90.43%), 11,112 Jamming records (4.83%), and 10,929 Battery Depletion records (4.75%). The same previously developed semi-synthetic dataset is reused in the present study, while the experimental partitioning and feature configuration described below are applied specifically for the current comparative evaluation.

The semi-synthetic augmentation and attack labelling were performed before the experimental partitioning. The resulting augmented dataset was subsequently divided using the device-aware splitting procedure described below, with device address used as the grouping variable. Consequently, all records associated with a given device were assigned exclusively to one of the training, validation, or test partitions, including its augmented records. Explicit overlap checks confirmed zero shared device identifiers across the three partitions, thereby preventing device-level leakage between training, validation, and test data.

For the experiments in this study, three attributes were selected as the input features: RSSI, altitude, and battery level. RSSI represents radio propagation behaviour observed at the receiver, altitude provides contextual information associated with the physical deployment of a device, and battery level represents device energy status. Together, these attributes provide complementary radio, contextual, and energy-related information. Although the augmented dataset also contains additional radio and temporal attributes, including SNR, SF, Battery Drop Speed (BDS), Event Step, and Time Rank, these attributes are not included in the feature vector used for the KD experiments reported in this study.

Unlike RSSI, altitude and battery level are device-related contextual attributes rather than direct radio measurements. The present study assumes their availability as represented in the evaluated semi-synthetic dataset and does not independently assess their integrity against malicious manipulation. Protecting or independently validating such device-reported attributes therefore remains an important consideration for operational deployment.

The input feature vector associated with the i -th communication record is therefore defined as follows:

$$X_i = [RSSI_i, Altitude_i, BL_i] \quad (1)$$

where $RSSI_i$, $Altitude_i$, and BL_i denote the received signal strength, altitude, and battery level associated with communication record i , respectively. The overall dataset preparation and feature-engineering procedure is illustrated in Figure 2.

To prevent device-specific information leakage, dataset partitioning was performed by device rather than by individual communication record. The 1921 devices were partitioned using a fixed device-aware split, with approximately 70% of the devices assigned to the training set and the remaining devices divided between the validation and test sets. This resulted in 1344 training devices, 288 validation devices, and 289 test devices. All communication records belonging to a particular device were retained within the same partition. Consequently, a device represented in the training set could not appear in either the validation or test set. The resulting record-level and class distributions are reported in Table 1.

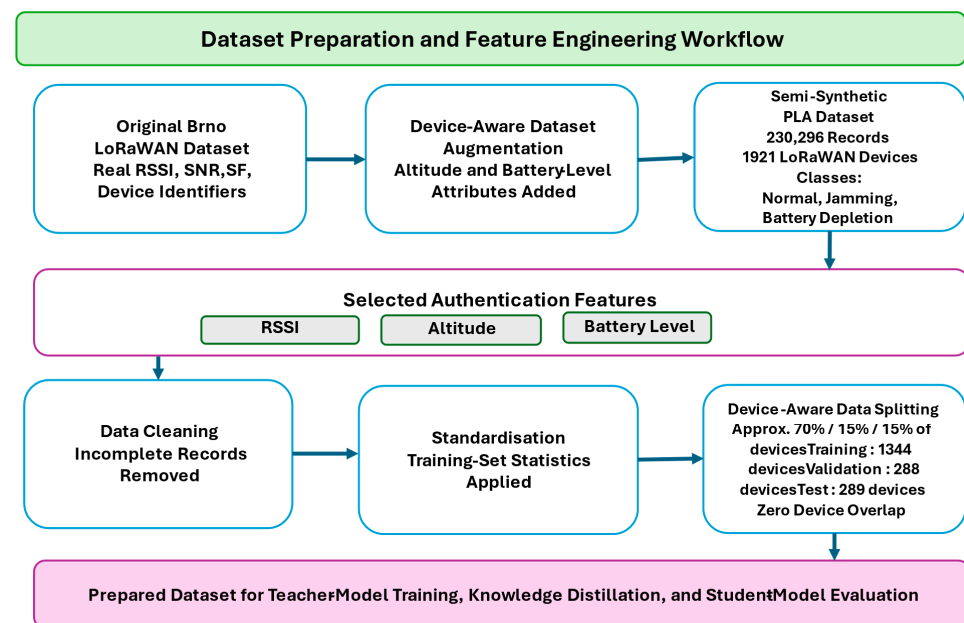


Figure 2. Dataset preparation and feature-engineering workflow, including semi-synthetic augmentation, feature selection, standardisation, and device-aware partitioning.

Table 1. Device-aware dataset partition and class distribution used in the experiments.

Partition	Records	Devices	Normal	Jamming	Battery Depletion
Training	177,221	1344	160,292 (90.45%)	8493 (4.79%)	8436 (4.76%)
Validation	28,040	288	25,329 (90.33%)	1391 (4.96%)	1320 (4.71%)
Test	25,035	289	22,634 (90.41%)	1228 (4.91%)	1173 (4.69%)
Overall	230,296	1921	208,255 (90.43%)	11,112 (4.83%)	10,929 (4.75%)

Explicit device-overlap checks confirmed zero overlap between the training and validation sets, zero overlap between the training and test sets, and zero overlap between the validation and test sets. This device-aware partitioning strategy provides a more stringent evaluation than random record-level splitting because the models are evaluated using communication records generated by devices that were not observed during training. It also reduces the risk that model performance is artificially inflated through the presence of records from the same physical device in both the training and evaluation partitions.

Following partitioning, the continuous input features were standardised using statistics calculated exclusively from the training partition. The resulting transformation parameters were subsequently applied unchanged to the validation and test partitions. Thus, neither validation nor test information contributed to the estimation of preprocessing parameters, preventing preprocessing-related information leakage. The same fixed device partitions and preprocessing procedure were used for the Teacher model and for the Direct, Hard-KD, and Soft-KD Student experiments, ensuring that differences between the evaluated training strategies were not caused by variations in the underlying training or evaluation data.

To further assess whether the classification performance depended disproportionately on any individual attribute, a feature-ablation analysis was conducted. The Teacher architecture was evaluated using each feature individually (RSSI, altitude, and battery level), each

pairwise combination (RSSI + altitude, RSSI + battery level, and altitude + battery level), and the complete three-feature configuration. Each configuration was evaluated using the same five random seeds as the main experiments, and performance was summarised using the mean and standard deviation. This analysis provides an empirical assessment of the individual and complementary contributions of the selected attributes. The corresponding results are presented in Section 4.

3.3. Teacher–Student Learning and Knowledge Distillation

The proposed approach employs a Teacher–Student learning strategy to investigate whether a compact neural network can retain the classification capability of a higher-capacity model. Both the Teacher and Student operate on the same three input features—RSSI, altitude, and battery level—and classify each communication record as Normal Communication, Jamming Attack, or Battery Depletion Attack.

The Teacher model is first trained offline using the ground-truth class labels. Once trained, the Teacher is used to provide supervisory information for the KD-based Student models. To distinguish the effect of knowledge transfer from the effect of model compression alone, three Student-training strategies are evaluated: Direct training, Hard-label Knowledge Distillation (Hard-KD), and Soft-label Knowledge Distillation (Soft-KD). The same Student architectures, device-aware data partitions, preprocessing procedure, and validation protocol are used across the three strategies, so that the primary difference is the source and form of supervision provided during Student training.

Direct Student Training. Direct training provides the non-KD baseline. The Student is trained directly using the original ground-truth class labels without receiving information from the Teacher. The training objective is as follows:

$$L_{Direct} = CE(z_S, y) \quad (2)$$

where L_{Direct} denotes the Direct Student training loss, CE denotes the cross-entropy loss function, z_S represents the output logits produced by the Student, and y denotes the original ground-truth class label. Direct training therefore measures the classification capability of the compressed Student architecture without knowledge transfer from the Teacher.

Hard-Label KD. In Hard-KD, the trained Teacher predicts a class for each training observation. The Teacher-generated hard label is obtained by selecting the class with the highest predicted probability. This Teacher prediction is denoted by y_T . The Student is then trained using y_T as its target instead of the original ground-truth label y . The Hard-KD training objective is as follows:

$$L_{HardKD} = CE(z_S, y_T) \quad (3)$$

where L_{HardKD} denotes the Hard-KD training loss, CE denotes the cross-entropy loss function, z_S represents the Student output logits, and y_T denotes the hard class label predicted by the trained Teacher model.

Thus, Direct training uses the original ground-truth label y , whereas Hard-KD uses the Teacher-generated label y_T . Hard-KD transfers the Teacher’s final classification decisions to the Student but does not preserve information contained in the Teacher’s probabilities for the non-selected classes.

Soft-Label KD. Soft-KD retains additional predictive information compared with Hard-KD by using the complete class-probability distribution produced by the Teacher. Following the conventional knowledge-distillation principle introduced by Hinton et al. [17], a temperature parameter is used to soften the probability distributions employed for knowledge transfer.

In the present implementation, the Teacher provides class probabilities rather than internal logits. Let p_T denote the original class-probability distribution produced by the Teacher. The temperature-scaled Teacher distribution is calculated as follows:

$$p_T^{(T)} = \text{softmax}\left(\frac{\log(p_T)}{T}\right) \quad (4)$$

where $p_T^{(T)}$ denotes the temperature-scaled Teacher probability distribution, (p_T) denotes the original Teacher class-probability distribution, T denotes the temperature parameter, $\log(\cdot)$ denotes the natural logarithm, and $\text{softmax}(\cdot)$ converts the scaled values into a probability distribution.

The corresponding temperature-scaled Student probability distribution is obtained from the Student logits:

$$p_S^{(T)} = \text{softmax}\left(\frac{z_S}{T}\right) \quad (5)$$

where $p_S^{(T)}$ denotes the temperature-scaled Student probability distribution, z_S represents the Student output logits, and T is the same temperature parameter applied to the Teacher and Student distributions.

A temperature greater than one produces a softer probability distribution, thereby exposing information about the Teacher's relative confidence across the three classes rather than transferring only its highest-probability decision [17].

The complete Soft-KD training objective is defined as follows:

$$L_{\text{SoftKD}} = \alpha L_{GT} + (1 - \alpha) T^2 L_{KD} \quad (6)$$

where L_{SoftKD} denotes the total Soft-KD training loss, L_{GT} denotes the loss calculated using the original ground-truth labels, L_{KD} denotes the knowledge-distillation loss between the Teacher and Student probability distributions, α controls the relative contribution of the two loss components, and T denotes the distillation temperature. The T^2 term provides the conventional temperature-related scaling used in KD [17].

The ground-truth component is as follows:

$$L_{GT} = CE(z_S, y) \quad (7)$$

where L_{GT} denotes the ground-truth classification loss, CE denotes cross-entropy, z_S represents the Student output logits, and y denotes the original ground-truth label.

The knowledge-distillation component is as follows:

$$L_{KD} = KL(p_T^{(T)}, p_S^{(T)}) \quad (8)$$

where L_{KD} denotes the knowledge-distillation loss, KL denotes the Kullback–Leibler divergence, $p_T^{(T)}$ denotes the temperature-scaled Teacher probability distribution, and $p_S^{(T)}$ denotes the temperature-scaled Student probability distribution. The KL -divergence term measures the difference between the Teacher and Student softened output distributions and encourages the Student to approximate the Teacher distribution.

For the experiments reported in this study, the Soft-KD parameters were set to $\alpha = 0.5$ and $T = 3.0$. Student optimisation was performed using the Adam optimiser with a learning rate of 0.001 and a batch size of 1024. Training was permitted for a maximum of 100 epochs, with early stopping using a patience of 10 epochs. At the end of each epoch, the Macro F1-score was evaluated on the fixed validation partition, and the model state achieving the highest validation Macro F1-score was retained. The same optimisation, early-stopping,

and model-selection procedures were applied consistently across Direct, Hard-KD, and Soft-KD Student training.

3.4. Teacher and Student Model Architectures

The proposed approach employs Teacher and Student models based on MLP architecture. Both models operate on the same three-feature input representation comprising RSSI, altitude, and battery level, as described in Section 3.2, and perform multi-class classification into three traffic categories: Normal Communication, Jamming Attack, and Battery Depletion. The use of a common input representation and output classification task ensures that differences in performance can be attributed to model capacity and training strategy rather than differences in the underlying feature space.

The Teacher is implemented using the scikit-learn MLP classifier with two hidden layers containing 64 and 32 neurons, respectively, and Rectified Linear Unit (ReLU) activation functions. Optimisation is performed using the Adam solver, with training permitted for a maximum of 500 iterations. For the repeated experiments, the Teacher random state is set to the corresponding experimental seed (42–46). The Teacher is trained using the original ground-truth class labels and the original imbalanced training distribution; no class weighting, oversampling, or undersampling is applied. No explicit early-stopping procedure is used for the Teacher, and optimisation terminates according to the convergence behaviour of the scikit-learn implementation or when the maximum number of iterations is reached. Once trained, the Teacher provides the supervisory information required by the Hard-KD and Soft-KD strategies described in Section 3.3 and is not required during Student inference.

The Student models are implemented as compact MLP architectures containing a single hidden layer. Three progressively smaller Student configurations are evaluated: Student (8), Student (4), and Student (2), containing 8, 4, and 2 hidden neurons, respectively. These configurations provide increasing levels of architectural compression relative to the Teacher and enable the relationship between model capacity and classification performance to be systematically investigated.

For each Student architecture, the training strategies described in Section 3.3 are applied under the same data-partitioning and preprocessing conditions. Direct training uses the original ground-truth labels and provides the non-KD reference for the corresponding Student architecture. Hard-KD trains the Student using the hard class labels predicted by the trained Teacher, whereas Soft-KD combines the original ground-truth supervision with information contained in the Teacher's softened class-probability distribution. This controlled comparison enables the effect of the Student architecture to be distinguished from the effect of knowledge transfer.

The reduction in network size substantially decreases the number of model parameters relative to the Teacher architecture. For an MLP with fully connected layers, the total number of trainable parameters is determined by the weights and biases connecting consecutive layers. Based on the three input features and three output classes used in this study, the Teacher (64, 32) architecture contains 2435 trainable parameters. In comparison, Student (8), Student (4), and Student (2) contain 59, 31, and 17 trainable parameters, respectively. Student (8), for example, therefore uses approximately 97.6% fewer trainable parameters than the Teacher while retaining the same three-feature input and three-class output structure.

The architectural reduction is intended to decrease model storage and computational requirements while preserving classification performance. However, architectural size alone does not establish practical efficiency. Accordingly, the experimental evaluation also measures serialised model size, training time, inference latency, and inference throughput.

These measurements are reported separately in Section 4 together with the classification results, allowing the performance–efficiency trade-off of the compressed Student models to be assessed empirically rather than inferred solely from parameter counts. The architectural configurations and corresponding trainable parameter counts of the Teacher and Student models are summarised in Table 2.

Table 2. Teacher and Student model architectures used in the experimental evaluation.

Model	Input Features	Hidden Layer Architecture	Output Classes	Trainable Parameters
Teacher MLP	3	(64, 32)	3	2435
Student (8)	3	(8)	3	59
Student (4)	3	(4)	3	31
Student (2)	3	(2)	3	17

4. Experimental Results and Evaluation

This section presents the experimental evaluation of the proposed lightweight PLA framework. The evaluation focuses on three complementary aspects. First, the classification performance of the Teacher and compressed Student architectures is examined under Direct, Hard-KD, and Soft-KD training to determine whether KD provides an advantage beyond architectural compression alone. Second, feature-ablation analysis is used to assess the individual and complementary contributions of the RSSI, altitude, and battery level. Third, the computational efficiency of the resulting models is evaluated in terms of model size, training time, inference latency, and throughput. Together, these experiments assess the trade-off between classification performance, model compression, training strategy, and computational efficiency.

4.1. Classification Performance

The classification performance of the proposed approach was evaluated using the fixed device-aware training, validation, and test partitions described in Section 3.2. The evaluation was designed to distinguish the effect of architectural compression from the effect of KD. Accordingly, the Student architectures were evaluated using Direct, Hard-KD, and Soft-KD training, while the higher-capacity Teacher model served as the reference classifier.

To account for variability arising from neural-network initialisation and stochastic optimisation, each Student configuration was evaluated across five random seeds (42, 43, 44, 45, and 46) while maintaining the same fixed device-aware data partitions. The results are therefore reported as the mean and standard deviation across the five runs. The classification performance was assessed using accuracy, Macro F1-score, Macro AUC-OVR, and class-specific recall. The Macro F1-score is particularly informative in the present evaluation because the dataset is strongly imbalanced, with Normal Communication accounting for approximately 90% of the observations.

Table 3 presents the classification performance of the Teacher and the three Student architectures under the different training strategies.

The Teacher achieved an Accuracy of 97.847%, a Macro F1-score of 91.007%, and a Macro AUC-OVR of 89.860%. Among the Student (8) configurations, all three training strategies produced performance close to the Teacher. The mean Macro F1-scores were $90.626 \pm 0.083\%$ for Soft-KD, $90.619 \pm 0.056\%$ for Direct training, and $90.591 \pm 0.069\%$ for Hard-KD. The differences among these three values are very small, providing no evidence of a meaningful performance advantage from KD over direct training for the 8-neuron Student.

The Teacher predictions differed from the original ground-truth labels for approximately 2.21% of the training samples. This relatively low disagreement indicates that Hard-KD largely reproduces the original supervision, with only a small proportion of training targets differing as a result of the Teacher’s decision boundary.

Table 3. Classification performance of the Teacher and Student models under Direct, Hard-KD, and Soft-KD training. Student Accuracy, Macro F1, and Macro AUC-OVR results are reported as mean $\pm$ standard deviation across five random seeds, while class-specific recall values are reported as means. For the Teacher, metrics with no observed variation across repeated evaluations are reported as mean values only, while Macro AUC-OVR is reported as mean $\pm$ standard deviation because variability was observed for this metric.

Model	Training	Accuracy (%)	Macro F1 (%)	Macro AUC-OVR (%)	Normal Recall (%)	Jamming Recall (%)	Battery Recall (%)
Teacher	Ground Truth	97.847	91.007	89.860 $\pm$ 0.228	99.912	76.026	80.853
Student (8)	Direct	97.760 $\pm$ 0.012	90.619 $\pm$ 0.056	89.495 $\pm$ 0.214	99.916	75.489	79.471
Student (8)	Hard-KD	97.754 $\pm$ 0.015	90.591 $\pm$ 0.069	89.241 $\pm$ 0.066	99.914	75.635	79.216
Student (8)	Soft-KD	97.762 $\pm$ 0.018	90.626 $\pm$ 0.083	89.608 $\pm$ 0.219	99.916	75.489	79.506
Student (4)	Direct	96.287 $\pm$ 3.283	78.881 $\pm$ 26.199	82.961 $\pm$ 14.460	99.930	60.456	63.495
Student (4)	Hard-KD	94.818 $\pm$ 4.022	67.093 $\pm$ 32.187	77.826 $\pm$ 15.929	99.946	45.440	47.553
Student (4)	Soft-KD	92.671 $\pm$ 4.503	55.523 $\pm$ 30.749	72.983 $\pm$ 15.323	99.222	29.772	32.123
Student (2)	Direct	92.618 $\pm$ 2.017	49.420 $\pm$ 16.221	74.488 $\pm$ 17.780	99.973	15.098	31.867
Student (2)	Hard-KD	93.358 $\pm$ 3.078	55.405 $\pm$ 24.719	74.446 $\pm$ 17.804	99.962	15.081	47.877
Student (2)	Soft-KD	91.877 $\pm$ 2.010	43.563 $\pm$ 16.306	70.870 $\pm$ 15.919	99.981	0.000	31.697

A different pattern is observed under stronger architectural compression. Student (4) exhibited substantially greater variability across the five random seeds. Direct training achieved a mean Macro F1-score of $78.881 \pm 26.199\%$, compared with $67.093 \pm 32.187\%$ for Hard-KD and $55.523 \pm 30.749\%$ for Soft-KD. The large standard deviations indicate that the 4-neuron architecture is considerably more sensitive to random initialisation and optimisation than Student (8).

Further compression to Student (2) resulted in a pronounced reduction in classification performance. Hard-KD produced the highest mean Macro F1-score among the Student (2) configurations at $55.405 \pm 24.719\%$, compared with $49.420 \pm 16.221\%$ for Direct training and $43.563 \pm 16.306\%$ for Soft-KD. Although Hard-KD produced the highest mean Macro F1-score within this highly compressed architecture, all Student (2) configurations remained substantially below the performance of Student (8) and exhibited considerable variability across runs. In particular, the class-specific results indicate a severe degradation in minority-class detection under the most aggressive compression settings.

To provide a more detailed assessment of the class-specific behaviour, Table 4 reports the precision, recall, F1-score, and false-positive rate (FPR) for the Teacher and the selected Student (8) configurations. These results complement the aggregate metrics in Table 3 and provide a direct assessment of minority-class detection and false-positive behaviour.

The detailed class-specific results show that the Teacher and all Student (8) variants maintain a very high precision and recall for Normal Communication, while the main performance reduction occurs in the minority Jamming and Battery Depletion classes. The three Student (8) training strategies exhibit closely comparable class-specific precision, recall, F1-score, and FPR values, reinforcing the aggregate results in Table 3 that neither Hard-KD nor Soft-KD provides a consistent advantage over Direct training. The higher FPR observed for the Normal class reflects its one-vs-rest formulation, in which the comparatively small Jamming and Battery Depletion populations constitute the negative class;

consequently, this value should not be interpreted as the overall false-alarm rate of the three-class classifier.

Table 4. Per-class classification performance of the Teacher and Student (8) models. Results are reported as mean $\pm$ standard deviation across five experimental runs.

Model	Training	Class	Precision (%)	Recall (%)	F1-Score (%)	FPR (%)
Teacher	Ground Truth	Normal	97.841 $\pm$ 0.000	99.912 $\pm$ 0.000	98.866 $\pm$ 0.000	20.783 $\pm$ 0.000
Teacher	Ground Truth	Jamming	98.419 $\pm$ 0.001	76.026 $\pm$ 0.045	85.785 $\pm$ 0.029	0.063 $\pm$ 0.000
Teacher	Ground Truth	Battery Depletion	97.432 $\pm$ 0.001	80.853 $\pm$ 0.047	88.371 $\pm$ 0.029	0.105 $\pm$ 0.000
Student (8)	Direct	Normal	97.741 $\pm$ 0.013	99.916 $\pm$ 0.000	98.816 $\pm$ 0.007	21.774 $\pm$ 0.126
Student (8)	Direct	Jamming	98.512 $\pm$ 0.000	75.489 $\pm$ 0.000	85.477 $\pm$ 0.000	0.059 $\pm$ 0.000
Student (8)	Direct	Battery Depletion	97.490 $\pm$ 0.008	79.471 $\pm$ 0.259	87.563 $\pm$ 0.160	0.101 $\pm$ 0.000
Student (8)	Hard-KD	Normal	97.735 $\pm$ 0.018	99.914 $\pm$ 0.002	98.813 $\pm$ 0.008	21.824 $\pm$ 0.174
Student (8)	Hard-KD	Jamming	98.473 $\pm$ 0.053	75.635 $\pm$ 0.203	85.556 $\pm$ 0.110	0.061 $\pm$ 0.002
Student (8)	Hard-KD	Battery Depletion	97.482 $\pm$ 0.005	79.216 $\pm$ 0.153	87.405 $\pm$ 0.095	0.101 $\pm$ 0.000
Student (8)	Soft-KD	Normal	97.742 $\pm$ 0.019	99.916 $\pm$ 0.000	98.817 $\pm$ 0.010	21.758 $\pm$ 0.190
Student (8)	Soft-KD	Jamming	98.512 $\pm$ 0.000	75.489 $\pm$ 0.000	85.477 $\pm$ 0.000	0.059 $\pm$ 0.000
Student (8)	Soft-KD	Battery Depletion	97.491 $\pm$ 0.012	79.506 $\pm$ 0.389	87.584 $\pm$ 0.241	0.101 $\pm$ 0.000

To further illustrate the class-specific error patterns, Figure 3 presents the confusion matrix for the directly trained Student (8) from a representative experimental run (seed 42). The matrix shows that Normal Communication is identified with a very high reliability, with only 19 of 22,634 Normal samples misclassified. The dominant residual errors arise from minority attack samples being classified as Normal, including 288 Jamming samples and 231 Battery Depletion samples. This behaviour is consistent with the class-specific recall values reported in Tables 3 and 4 and reflects the challenge posed by the strongly imbalanced class distribution.

		Predicted Class		
		Normal	Jamming	Battery Depletion
Actual Class	Normal	22,615	8	11
	Jamming	288	927	13
	Battery Depletion	231	6	936

Student (8), Direct training, representative run (seed 42)

Figure 3. Confusion matrix for the directly trained Student (8) model from the representative experimental run using seed 42. Rows indicate the ground-truth classes and columns indicate the predicted classes.

The relationship between Student capacity, training strategy, and run-to-run variability is illustrated in Figure 4.

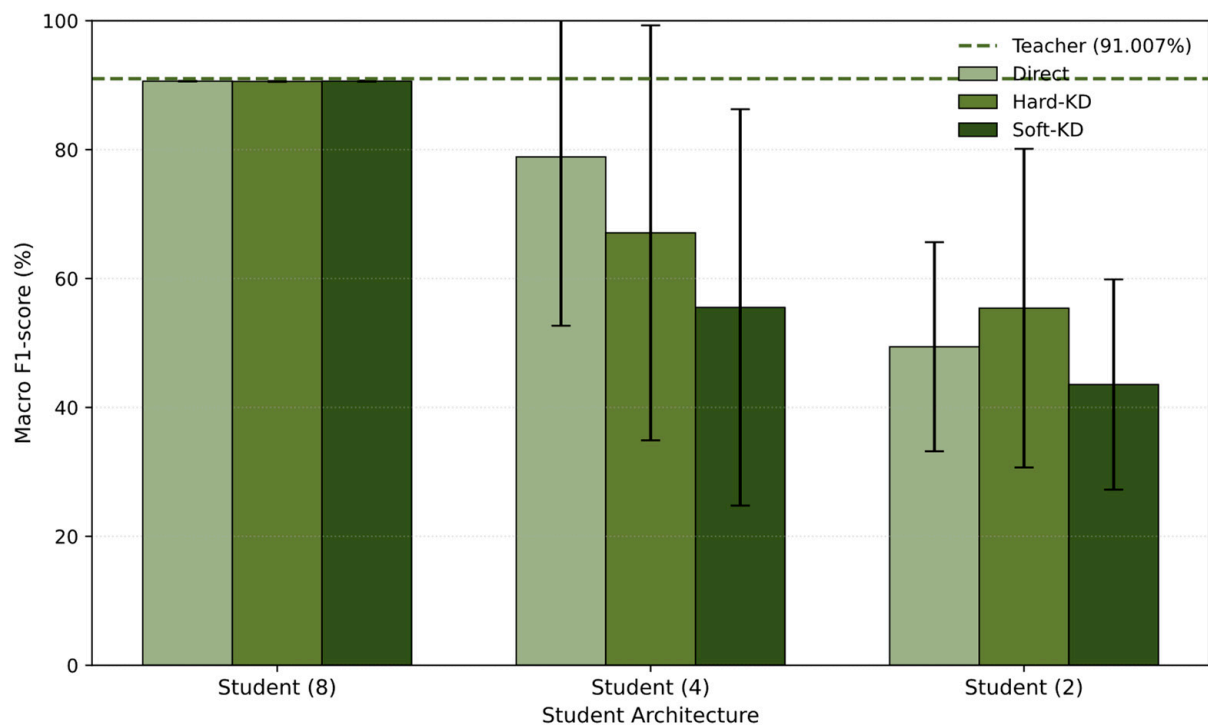


Figure 4. Macro F1-score comparison of Direct, Hard-KD, and Soft-KD training across the Student architectures. Bars represent the mean performance across five random seeds (42, 43, 44, 45, and 46), and error bars indicate $\pm$ one standard deviation. The dashed horizontal line represents the Teacher Macro F1-score (91.007%).

Figure 4 highlights two distinct behaviours. First, Student (8) remains close to the Teacher reference under all three training strategies, and its very small error bars demonstrate stable performance across the five random seeds. The marginal differences among Direct, Hard-KD, and Soft-KD indicate that the 8-neuron architecture possesses sufficient capacity to learn the classification task effectively even without KD.

Second, the substantially larger error bars observed for Student (4) and Student (2) demonstrate an increasing run-to-run instability as the model capacity is reduced. This suggests that aggressive architectural compression, rather than the choice of distillation strategy alone, becomes the dominant limitation for the smallest Student models. KD therefore does not consistently compensate for the insufficient Student capacity. Among the evaluated architectures, Student (8) provides the most favourable balance between classification performance, stability, and architectural compression.

4.2. Feature-Ablation Analysis

To assess the contribution of the selected input attributes, a feature-ablation analysis was conducted using the Teacher architecture. As described in Section 3.2, the model was evaluated using each feature individually, each pairwise feature combination, and the complete three-feature configuration comprising the RSSI, altitude, and battery level. The same fixed device-aware training, validation, and test partitions were maintained across all configurations, and each experiment was repeated across five random seeds (42, 43, 44, 45, and 46). The results are presented in Table 5.

The feature-ablation results reveal distinct and complementary contributions from the RSSI and battery level. The RSSI alone provided a useful discrimination of Jamming attacks, achieving a Jamming recall of 67.573%, but failed to identify Battery Depletion attacks. Conversely, battery level alone achieved a Battery Depletion recall of 69.548% but produced zero recall for Jamming attacks. These results indicate that the RSSI primarily

captures information associated with anomalous radio behaviour, whereas the battery level captures information associated with abnormal energy depletion.

Altitude alone provided little discriminatory capability for the attack classes. Although the Altitude-only configuration achieved 100% Normal recall, it produced zero recall for both Jamming and Battery Depletion attacks. This result should not be interpreted as a strong overall classification performance; rather, it reflects the model's tendency to classify observations as the dominant Normal class when altitude is used in isolation. The corresponding Macro F1-score of 31.654% further demonstrates that altitude alone provides insufficient information for distinguishing the three traffic classes.

Table 5. Feature-ablation performance of the Teacher architecture using individual, pairwise, and complete feature configurations. Results are reported as mean $\pm$ standard deviation across five random seeds.

Feature Configuration	Accuracy (%)	Macro F1 (%)	Macro AUC-OVR (%)	Normal Recall (%)	Jamming Recall (%)	Battery Recall (%)
RSSI only	93.653 $\pm$ 0.057	58.781 $\pm$ 0.314	72.496 $\pm$ 0.348	99.921 $\pm$ 0.038	67.573 $\pm$ 1.630	0.000 $\pm$ 0.000
Altitude only	90.409 $\pm$ 0.000	31.654 $\pm$ 0.000	50.009 $\pm$ 0.752	100.000 $\pm$ 0.000	0.000 $\pm$ 0.000	0.000 $\pm$ 0.000
Battery only	93.562 $\pm$ 0.121	59.036 $\pm$ 0.461	73.037 $\pm$ 0.202	99.883 $\pm$ 0.174	0.000 $\pm$ 0.000	69.548 $\pm$ 2.260
RSSI + Altitude	93.647 $\pm$ 0.026	58.735 $\pm$ 0.082	71.800 $\pm$ 0.772	99.939 $\pm$ 0.052	67.134 $\pm$ 0.661	0.000 $\pm$ 0.000
RSSI + Battery	97.766 $\pm$ 0.009	90.657 $\pm$ 0.035	89.848 $\pm$ 0.274	99.912 $\pm$ 0.000	76.059 $\pm$ 0.115	79.096 $\pm$ 0.111
Altitude + Battery	93.730 $\pm$ 0.042	59.896 $\pm$ 0.110	73.180 $\pm$ 0.207	99.842 $\pm$ 0.087	0.000 $\pm$ 0.000	73.913 $\pm$ 0.904
RSSI + Altitude + Battery	97.845 $\pm$ 0.002	90.997 $\pm$ 0.009	89.759 $\pm$ 0.155	99.911 $\pm$ 0.002	76.026 $\pm$ 0.045	80.818 $\pm$ 0.000

The pairwise feature combinations provide further evidence of the complementary relationship between the RSSI and battery level. Combining altitude with RSSI did not materially improve the ability to detect Battery Depletion attacks, while combining the altitude with battery level continued to provide no effective discrimination of Jamming attacks. In contrast, the RSSI + Battery configuration detected both attack classes, achieving a Jamming and Battery Depletion recall of 76.059% and 79.096%, respectively. Its Macro F1-score increased substantially to 90.657 $\pm$ 0.035%, demonstrating that the combination of radio and energy-related information provides most of the discriminatory capability required for the three-class classification task.

The complete RSSI + Altitude + Battery configuration achieved the highest overall Macro F1-score of 90.997 $\pm$ 0.009% and increased the Battery Depletion recall to 80.818%. However, the improvement relative to RSSI + Battery was modest. RSSI + Battery also achieved a marginally higher Macro AUC-OVR and Jamming recall than the complete feature set. These findings indicate that the RSSI and battery level constitute the principal complementary attributes in the proposed framework, while altitude provides a smaller contextual contribution when incorporated into the complete three-feature representation.

The effect of feature selection on classification performance is further illustrated in Figure 5.

Figure 5 further highlights the substantial performance improvement obtained when the RSSI and battery level are used jointly. Configurations that lack either of these two attributes exhibit markedly lower Macro F1-scores, whereas RSSI + Battery approaches the performance of the complete three-feature configuration. The addition of altitude provides a further but comparatively small improvement in Macro F1-score. Overall, the ablation results show that the complete three-feature representation provides the highest Macro F1-score, while demonstrating that the principal discriminatory information is provided by the complementary RSSI and battery-level attributes and that altitude contributes only a modest additional benefit.

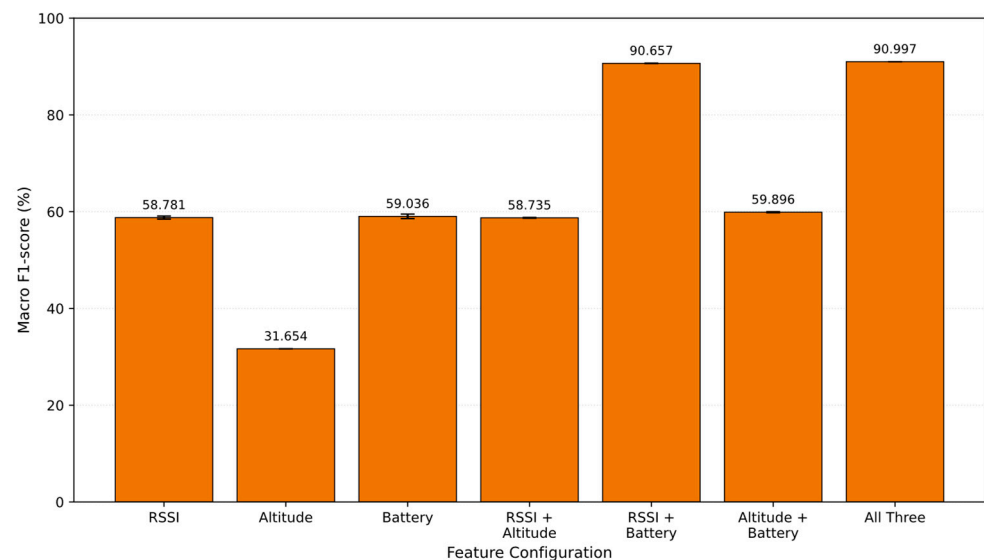


Figure 5. Macro F1-score obtained using individual, pairwise, and complete feature configurations. Bars represent the mean performance across five random seeds (42–46), and error bars indicate $\pm$ one standard deviation.

4.3. Computational Efficiency

In addition to the classification performance, the computational efficiency of the proposed approach was evaluated to determine whether architectural compression produces measurable reductions in the model complexity and inference requirements. The Teacher and Student (8) models were compared in terms of the trainable parameters, serialised model size, training time, single-sample inference latency, batch-derived inference latency, and throughput. Student (8) was selected for this comparison because the results in Section 4.1 identified it as the most stable compressed architecture, retaining a classification performance close to that of the Teacher across all three training strategies.

All efficiency measurements were obtained on the same CPU-based experimental platform, equipped with a 13th Gen Intel Core i7-1355U processor (1.70 GHz) and 16 GB of RAM and running 64-bit Windows 11 Education. The experiments were implemented using Python 3.12.10, PyTorch 2.13.0+cpu, and scikit-learn 1.6.0. The single-sample inference latency was measured following 100 warm-up runs and averaged over 1000 timed repetitions. The batch inference was evaluated following five warm-up runs and averaged over 30 repetitions using the complete test set. For the batch inference, the complete test partition was processed in a single prediction call for each repetition, and the reported per-sample batch latency was calculated by dividing the mean elapsed inference time by the number of test samples. The throughput was derived from the same full-test-set timing measurements. These repeated measurements were used to reduce the influence of the transient timing variation. The resulting efficiency measurements are presented in Table 6.

The Student (8) architecture contains only 59 trainable parameters compared with 2435 parameters in the Teacher, corresponding to a parameter reduction of approximately 97.6%. Similarly, the serialised Student model size was approximately 2.80 KB compared with 46.65 KB for the Teacher, representing a reduction of approximately 94.0%. Because Direct, Hard-KD, and Soft-KD use the same Student (8) architecture, their parameter counts and serialised model sizes are identical. These reductions therefore result from architectural compression rather than from the specific Student-training strategy.

The reported model size represents the serialised storage size and should not be interpreted as runtime memory consumption. Runtime memory was not treated as a primary quantitative comparison in the present evaluation because process-level memory

measurements obtained on the host system do not reliably isolate the memory attributable to an individual model. Accordingly, the serialised model size is reported as a storage requirement, while precise model-specific runtime memory profiling on the intended deployment platform remains for future evaluation.

Table 6. Computational-efficiency comparison of the Teacher and Student (8) models.

Model	Parameters	Training Time (s)	Model Size (KB)	Single-Sample Latency (ms)	Batch Latency (ms/Sample)	Throughput (Samples/s)
Teacher	2435	42.70	46.65	0.265 ± 0.144	0.004219	237,036
Student (8) Direct	59	158.54	2.80	0.596 ± 1.068	0.000347	2,881,320
Student (8) Hard-KD	59	133.68	2.80	0.224 ± 0.353	0.000234	4,270,433
Student (8) Soft-KD	59	88.31	2.80	0.162 ± 0.121	0.000180	5,568,576

The batch-inference measurements demonstrate a substantially higher processing efficiency for the compressed Student architecture under the evaluated CPU-based implementation. The Teacher required approximately 0.004219 ms per sample during the batch inference and achieved a throughput of approximately 237,036 samples/s. In comparison, the Student (8) variants achieved batch-derived latencies between approximately 0.000180 and 0.000347 ms per sample and throughputs between approximately 2.88 and 5.57 million samples/s. These measurements indicate a substantially greater batch-processing efficiency for the compact Student architecture under the evaluated CPU environment.

The single-sample latency results require more cautious interpretation. Student (8) Soft-KD and Hard-KD exhibited a lower mean single-sample latency than the Teacher, whereas Student (8) Direct produced a higher measured latency. Moreover, the relatively large timing variability, particularly for Direct training, indicates that very short single-sample measurements are sensitive to runtime overhead and system-level timing effects. In addition, the Teacher implementation uses scikit-learn, whereas the Student implementation uses PyTorch; consequently, these latency differences cannot be attributed exclusively to the neural-network architecture. For this reason, the parameter count, serialised model size, batch-derived latency, and throughput provide the more consistent evidence of computational reduction.

The training time exhibits a different pattern. The Teacher required approximately 42.70 s to train, whereas the Student (8) variants required approximately 88.31–158.54 s depending on the training strategy. Thus, the compact Student does not provide a reduction in offline training time in the present implementation. This result does not conflict with the objective of the proposed approach because the training time is influenced by the implementation framework, optimisation procedure, and, for the KD variants, the additional operations associated with Teacher supervision. The primary efficiency objective is to reduce the complexity and inference requirements of the resulting Student model rather than the duration of offline model optimisation. The Teacher and distillation procedures are required only during offline training and are not required once the Student model has been trained.

The relative reductions in model-storage requirements and the measured improvements in batch-inference throughput are illustrated in Figure 6. For clarity, both measures are normalised relative to the Teacher model.

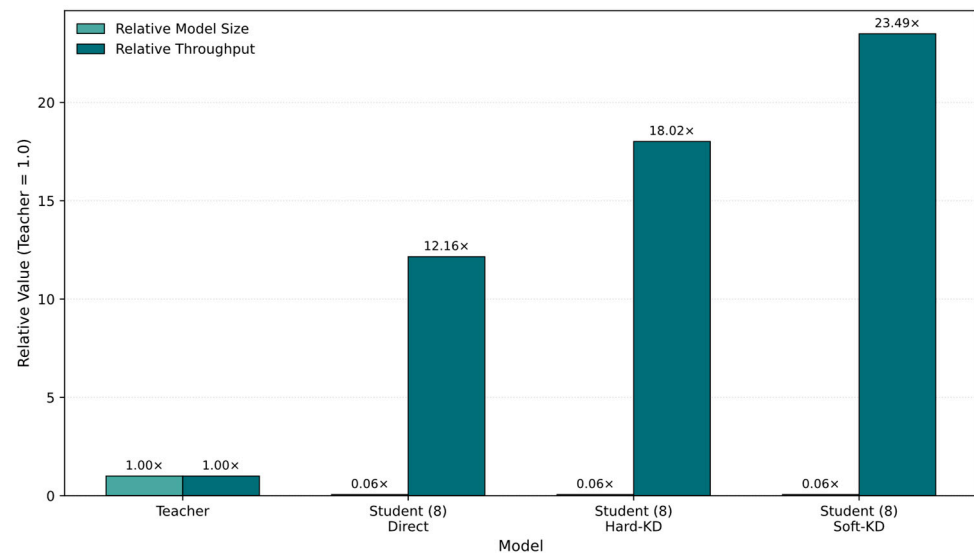


Figure 6. Relative serialised model size and batch-inference throughput of the Teacher and Student (8) models under the evaluated CPU environment. Values are normalised relative to the Teacher ($1.0\times$). All Student (8) variants require approximately 6% of the Teacher’s serialised model size, while exhibiting substantially higher measured batch throughput.

As shown in Figure 6, all Student (8) variants require only approximately $0.06\times$ the serialised storage of the Teacher, reflecting the substantial reduction in architectural complexity. The measured batch throughput was approximately $12.16\times$, $18.02\times$, and $23.49\times$ the Teacher throughput for Direct, Hard-KD, and Soft-KD, respectively. Since all three Student variants share the same architecture, the differences in the measured throughput among the Student training strategies should not be interpreted as structural advantages of one KD strategy over another. Rather, the principal finding is that the compressed Student architecture has substantially lower serialised storage requirements and, across the evaluated Student variants, exhibits a considerably higher measured batch-processing throughput than the Teacher under the evaluated CPU-based implementation.

Overall, the efficiency results demonstrate that Student (8) provides substantial reductions in the parameter count and serialised model size while retaining a classification performance close to that of the Teacher, as reported in Section 4.1. The batch-inference measurements further indicate a substantially higher processing throughput under the evaluated experimental environment. However, these measurements represent software-based CPU benchmarking rather than deployment on operational LoRaWAN hardware. Consequently, they provide evidence of computational efficiency under the experimental platform but should not be interpreted as direct measurements of gateway- or end-device-level deployment performance.

5. Conclusions

This paper investigated architectural compression and alternative training strategies for lightweight multi-attribute PLA in LoRaWAN networks. Building upon our previously developed PLA framework, the study examined whether progressively smaller Student models could preserve the classification capability of a higher-capacity Teacher and, importantly, whether Teacher supervision through Hard-KD or Soft-KD provides an advantage over directly training the same compact architectures. The study retained the RSSI, altitude, and battery level as the multi-attribute input features for distinguishing Normal Communication, Jamming Attack, and Battery Depletion Attack traffic.

The experimental results demonstrated that substantial architectural compression could be achieved while retaining a classification performance close to that of the Teacher

with the Student (8) architecture. Its Macro F1-scores were $90.619 \pm 0.056\%$ for Direct training, $90.591 \pm 0.069\%$ for Hard-KD, and $90.626 \pm 0.083\%$ for Soft-KD, compared with 91.007% for the Teacher. In contrast, the more aggressively compressed Student (4) and Student (2) architectures exhibited substantially greater variability across random seeds and reduced the minority-class detection performance, indicating that excessive compression can compromise the model stability and attack discrimination.

A key finding of this study was that KD did not consistently outperform the direct training of the equivalent compact Student architectures. For Student (8), Direct, Hard-KD, and Soft-KD produced a very similar classification performance, while the more compressed architectures did not exhibit a consistent advantage from Teacher supervision. These results indicate that much of the retained performance can be attributed to the representational capacity of the compact architecture itself rather than to knowledge transfer alone. The inclusion of a Direct-training baseline was therefore essential for distinguishing the effect of architectural compression from the contribution of KD.

The feature-ablation analysis further demonstrated that the RSSI and battery level provide the principal complementary discriminatory information for the evaluated three-class task. The RSSI was particularly informative for Jamming detection, whereas the battery level was important for Battery Depletion detection. Altitude alone provided little discriminatory capability for the attack classes, although its inclusion with the RSSI and battery level produced a small improvement in the Macro F1-score. These results show that the complete three-feature representation achieved the highest Macro F1-score, while confirming that the RSSI and battery level supplied the dominant discriminatory information and that the altitude contributed a comparatively modest additional benefit.

From a computational perspective, Student (8) reduced the number of trainable parameters from 2435 to 59, corresponding to a reduction of approximately 97.6%, while reducing the serialised model size from 46.65 KB to 2.80 KB, or approximately 94.0%. CPU-based batch-inference measurements also showed a substantially lower per-sample latency and higher throughput for the compact Student architecture than for the Teacher under the evaluated implementation. However, the Student models did not reduce the offline training time in the present implementation, and differences in the runtime frameworks between the scikit-learn Teacher and PyTorch Students limit the direct attribution of the measured latency differences solely to architectural compression.

Overall, the results demonstrate that lightweight Student architectures can substantially reduce the model complexity while preserving a strong classification performance within the evaluated multi-attribute PLA framework for LoRaWAN. At the same time, the controlled comparison among Direct, Hard-KD, and Soft-KD demonstrates that KD should not be assumed to improve a compact model when the underlying classification task can already be learned effectively through direct supervision. The reported efficiency measurements were obtained through software-based CPU benchmarking rather than deployment on operational LoRaWAN hardware. Future work should therefore evaluate the compressed models on representative Network Server, gateway, and embedded platforms, investigate additional compression techniques such as quantisation and pruning, and assess robustness under broader network conditions, device populations, and attack scenarios.

Author Contributions: Conceptualisation, A.P. and A.M.; methodology, A.P.; software, A.P.; validation, A.P.; formal analysis, A.P.; investigation, A.P.; data curation, A.P.; visualisation, A.P.; writing—original draft preparation, A.P.; writing—review and editing, A.P. and A.M.; supervision, A.M., R.K., A.T. and I.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The complete processed semi-synthetic dataset used in this study is publicly available on Zenodo at <https://doi.org/10.5281/zenodo.22044142> (accessed on 21 August 2026). The dataset was developed from the publicly available LoRaWAN Traffic Analysis Dataset using the device-aware semi-synthetic augmentation and attack-modelling procedures described in Section 3.2 and in our previous work [12].

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

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
AUC-OVR	Area Under the Receiver Operating Characteristic Curve, One-vs-Rest
BDS	Battery Drop Speed
BL	Battery Level
CSI	Channel State Information
DL	Deep Learning
IoT	Internet of Things
KD	Knowledge Distillation
LPWAN	Low-Power Wide-Area Network
LoRaWAN	Long Range Wide Area Network
ML	Machine Learning
MLP	Multi-Layer Perceptron
NS	Network Server
PLA	Physical Layer Authentication
ReLU	Rectified Linear Unit
RFF	Radio Frequency Fingerprinting
RSSI	Received Signal Strength Indicator
SF	Spreading Factor
SNR	Signal-to-Noise Ratio
TinyML	Tiny Machine Learning

References

1. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [\[CrossRef\]](#)
2. LoRa Alliance. *LoRaWAN® Specification v1.1*; LoRa Alliance: Fremont, CA, USA, 2017.
3. Adelantado, F.; Vilajosana, X.; Tuset-Peiró, P.; Martinez, B.; Melia-Seguí, J.; Watteyne, T. Understanding the Limits of LoRaWAN. *IEEE Commun. Mag.* **2017**, *55*, 34–40. [\[CrossRef\]](#)
4. Mekki, K.; Bajic, E.; Chaxel, F.; Meyer, F. A Comparative Study of LPWAN Technologies for Large-Scale IoT Deployment. *ICT Express* **2019**, *5*, 1–7. [\[CrossRef\]](#)
5. Aras, E.; Ramachandran, G.S.; Lawrence, P.; Hughes, D. Exploring the Security Vulnerabilities of LoRa. In Proceedings of the 2017 IEEE 3rd International Conference on Cybernetics (CYBCONF), Exeter, UK, 21–23 June 2017; pp. 1–6. [\[CrossRef\]](#)
6. Butun, I.; Pereira, N.; Gidlund, M. Analysis of LoRaWAN v1.1 Security. In Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects (SMARTOBJECTS'18), Los Angeles, CA, USA, 25 June 2018; pp. 1–6. [\[CrossRef\]](#)
7. Xie, N.; Li, Z.; Tan, H. A Survey of Physical-Layer Authentication in Wireless Communications. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 282–310. [\[CrossRef\]](#)
8. Alhoraibi, L.; Alghazzawi, D.; Alhebshi, R.; Rabie, O.B.J. Physical Layer Authentication in Wireless Networks-Based Machine Learning Approaches. *Sensors* **2023**, *23*, 1814. [\[CrossRef\]](#) [\[PubMed\]](#)
9. Meng, R.; Xu, B.; Xu, X.; Sun, M.; Wang, B.; Han, S.; Lv, S.; Zhang, P. A Survey of Machine Learning-Based Physical-Layer Authentication in Wireless Communications. *J. Netw. Comput. Appl.* **2025**, *235*, 104085. [\[CrossRef\]](#)
10. Meng, R.; Cheng, X.; Gao, S.; Xu, X.; Dong, C.; Nan, G.; Tao, X.; Zhang, P.; Quek, T.Q.S. Generative AI for Physical-Layer Authentication. *IEEE Wirel. Commun.* 2026; *in press*. [\[CrossRef\]](#)
11. Pourghasem, A.; Kirner, R.; Tsokanos, A.; Mporas, I.; Mylonas, A. Machine Learning and Deep Learning-Based Multi-Attribute Physical-Layer Authentication for Spoofing Detection in LoRaWAN. *Future Internet* **2025**, *17*, 68. [\[CrossRef\]](#)

12. Pourghasem, A.; Kirner, R.; Tsokanos, A.; Mporas, I.; Mylonas, A. Multi-Attribute Physical-Layer Authentication Against Jamming and Battery-Depletion Attacks in LoRaWAN. *Future Internet* **2026**, *18*, 38. [\[CrossRef\]](#)
13. Cordova-Cardenas, R.; Amor, D.; Gutiérrez, Á. Edge AI in Practice: A Survey and Deployment Framework for Neural Networks on Embedded Systems. *Electronics* **2025**, *14*, 4877. [\[CrossRef\]](#)
14. Merenda, M.; Porcaro, C.; Iero, D. Edge Machine Learning for AI-Enabled IoT Devices: A Review. *Sensors* **2020**, *20*, 2533. [\[CrossRef\]](#) [\[PubMed\]](#)
15. Trigkas, A.; Piromalis, D.; Papageorgas, P. Edge Intelligence in Urban Landscapes: Reviewing TinyML Applications for Connected and Sustainable Smart Cities. *Electronics* **2025**, *14*, 2890. [\[CrossRef\]](#)
16. Heydari, S.; Mahmoud, Q.H. Tiny Machine Learning and On-Device Inference: A Survey of Applications, Challenges, and Future Directions. *Sensors* **2025**, *25*, 3191. [\[CrossRef\]](#) [\[PubMed\]](#)
17. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. In Proceedings of the NIPS Deep Learning and Representation Learning Workshop, Montreal, QC, Canada, 7–12 December 2015. [\[CrossRef\]](#)
18. Shehzadi, T.; Noor, R.; Ifza, I.; Liwicki, M.; Stricker, D.; Afzal, M.Z. Knowledge Distillation in Object Detection: A Survey from CNN to Transformer. *Sensors* **2026**, *26*, 292. [\[CrossRef\]](#) [\[PubMed\]](#)
19. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [\[CrossRef\]](#)
20. Hung, C.-W.; Tsai, C.-Y.; Wang, C.-C.; Lee, C.-H. TinyFL_HKD: Enhancing Edge AI Federated Learning with Hierarchical Knowledge Distillation Framework. *IEEE Sens. J.* **2025**, *25*, 12038–12049. [\[CrossRef\]](#)
21. Akram, M.A.; Mian, A.N.; Biswas, A.K.; Kumari, S.; Chen, C.-M. Privacy-Preserving Lightweight LoRaWAN Authentication Protocol for IoT Applications. *IEEE Internet Things J.* **2025**, *12*, 42790–42802. [\[CrossRef\]](#)
22. Ruotsalainen, H.; Shen, G.; Zhang, J.; Fujdiak, R. LoRaWAN Physical Layer-Based Attacks and Countermeasures, A Review. *Sensors* **2022**, *22*, 3127. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Hamdaoui, B.; Elmaghrib, A. Deep-Learning-Based Device Fingerprinting for Increased LoRa-IoT Security: Sensitivity to Network Deployment Changes. *IEEE Netw.* **2022**, *36*, 204–210. [\[CrossRef\]](#)
24. Jagannath, A.; Jagannath, J.; Kumar, P.S.P.V. A Comprehensive Survey on Radio Frequency Fingerprinting: Traditional Approaches, Deep Learning, and Open Challenges. *Comput. Netw.* **2022**, *219*, 109455. [\[CrossRef\]](#)
25. Butun, I.; Pereira, N.; Gidlund, M. Security Risk Analysis of LoRaWAN and Future Directions. *Future Internet* **2019**, *11*, 3. [\[CrossRef\]](#)
26. Pourghasem, A. Semi-Synthetic LoRaWAN Dataset for Jamming and Battery-Depletion Attack Detection; Version 1.0.0; Zenodo. 2026. Available online: <https://zenodo.org/records/22044142> (accessed on 21 August 2026).
27. Povalac, A.; Kral, J. LoRaWAN Traffic Analysis Dataset. Zenodo. 2023. Available online: <https://zenodo.org/records/7919213> (accessed on 21 August 2026).

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.