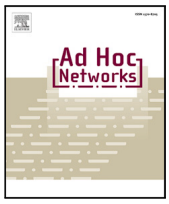
Ad Hoc Networks 183 (2026) 104101

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



Protecting autonomous systems from GPS spoofing with a machine learning-driven approach

Arslan Shafique [a], Abid Mehmood [b,*], Moatsum Alawida [b], Shehzad Ashraf Chaudhry [b,c]

[a] School of Electronic and Nanoscale Engineering, University of Glasgow, Glasgow, G12 8QQ, United Kingdom
[b] Department of Computer Science and Information Technology, College of Engineering Abu Dhabi University, Abu Dhabi, 59911, United Arab Emirates
[c] Department of Software Engineering, Faculty of Engineering and Architecture, Nisantasi University, Istanbul, Turkey

ARTICLE INFO

Keywords:
UAVs
Mathematical modeling
Spoofing
Security
GPS
Cyber threats
Machine learning

ABSTRACT

With the rapid evolution of interactive multimedia systems, ensuring strong security measures has become increasingly vital. Autonomous platforms, such as drones, are vulnerable to sophisticated cyber threats, including jamming and spoofing attacks. One common spoofing strategy involves manipulating Global Positioning System (GPS) signals. By broadcasting counterfeit signals, attackers can deceive drone navigation systems. To mitigate these risks, this research introduces a machine learning-based framework aimed at intelligently detecting spoofing attempts. The approach employs signal characteristics that reflect variations in jitter, shimmer, and frequency modulations, rooted in mathematical analysis. A private dataset is used for this work. This dataset, developed by our research team, is collected under varied environmental conditions, such as during daylight and in low-light settings, over multiple sessions. It categorizes signal statistics into three distinct ranges: the initial and final segments suggest spoofed signals, whereas the middle range corresponds to genuine ones. Further, using signal reception strength (SRS) values, additional data was sourced from trusted Long Range Wide Area Network (LoRaWAN) devices. This information supports the development of a singular-class support vector machine (S-CSVM) classifier for spoofing detection. For training purposes, the dataset was partitioned into two subsets: a training set (T_{train}) and a testing set (T_{test}). The performance of the model is evaluated using standard metrics, including precision, recall, F-score, and overall accuracy. By utilizing all available features, the model achieves its highest scores: 99.99% precision, 99.77% recall, and 99.95% F-score. The highest accuracy of 99.22% is achieved when all features are selected, and the distance between LoRaWAN devices and the monitoring device ranges from 6 to 8 m, outperforming the results obtained through feature selection in the ablation study. A thorough evaluation further highlights how the proposed ML-based solution outperforms existing methods.

1. Introduction

Nowadays, self-piloting systems, such as Unmanned Aerial Vehicles (UAVs) or Drones, are increasingly utilized for various purposes, including aerial surveillance, secure communication, and packet delivery [1]. However, in hazardous environments, communication signals exchanged between Drones and ground stations are susceptible to loss or corruption due to potential cyber-attacks like spoofing, and jamming [2–6]. Ensuring a safe landing or executing a return-to-home approach in such critical situations creates significant challenges. The existing autopilot systems often lack robustness against potent cybersecurity threats, that makes safe landings in unforeseen areas a challenging task. Therefore, there is a need for an intelligent system for Drones that can be able to classify the spoofed and original signals and dynamically respond to mitigate threats with minimal computational overhead.

Securing Drones presents several challenges, such as the constraints of lightweight, small size, and limited computational capabilities [7, 8]. The intricate architecture of Drones further complicates efforts to enhance their security. Nonetheless, achieving robust Drones security entails striking a balance between robust protection and considerations such as expected benefits, cost, and functionalities. For a comprehensive understanding of Drone architecture, Pohl et al. [9] provide detailed insights in their work.

To effectively defend Drones against cyber-attacks, it is necessary that Drones must be equipped with the robust architecture [10]. Drones face vulnerabilities in two primary forms: Denial of Service (DoS)

* Corresponding author.
E-mail address: abid.mehmood@adu.ac.ae (A. Mehmood).

https://doi.org/10.1016/j.adhoc.2025.104101
Received 7 June 2025; Received in revised form 8 October 2025; Accepted 24 November 2025
Available online 26 November 2025

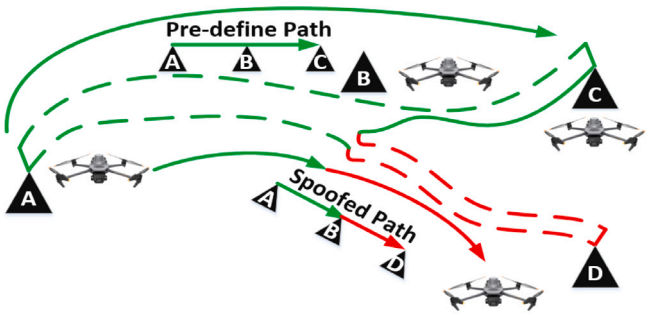


Fig. 1. A pre-defined and a spoofed paths.

attacks [11], and integrity attacks [12]. Integrity attacks encompass activities such as false data injection, and signal spoofing, while DoS attacks include jamming, black hole, and gray attacks.

The architecture and the system of the Drone rely on the Global Positioning System (GPS) for accurate navigation. Despite its effectiveness in controlling Drones, GPS is susceptible to the interference of the Radio Frequency (RF) [13,14]. Jamming and GPS spoofing represent two significant vulnerabilities that creates substantial threats to both military and civilian GPS users. Jamming is used to transmit a radio signal having relatively high frequency that masks the legitimate signal with noise to disrupt its reception [15–17]. Conversely, GPS spoofing requires an attacker to send fake signals to generate false positions or modify predefined positions to misguide the Drone system and divert it from its intended mission. Therefore, civilian or military Drones can be manipulated to deviate from their predetermined trajectories without the awareness of the users. Fig. 1 illustrates two trajectories: the predefined path ABC and the spoofed trajectory ABD. Upon successful execution of a spoofing attack, the Drone deviates from point B toward point D, divergent to its desired destination.

In recent times, security analysts have observed that Drones can be easily manipulated through GPS spoofing attacks when utilizing inexpensive software and hardware [18–20]. Drawing from vulnerabilities outlined in Jafarnia-Jahromi et al. [21], Shepard et al. [22], Ranyal and Jain [23], GPS spoofing can be classified into three primary categories: (i) GPS signal simulators, (ii) GPS receiver-based spoofers, and (iii) advanced receiver-based spoofers [24]. In the first category, simulators are employed to transmit GPS signals in conjunction with radio signals that generates a replicated GPS signal without the need for blending it with noise. In the second category, GPS receivers synchronize with spoofing transmitters to identify the predefined location. Detecting this type of spoofing proves challenging. The third category represents a more sophisticated approach. Here, It is presumed that the position and the velocity of the affected receiver are precisely identified. Identifying this form of attack using conventional anti-spoofing measures is nearly impossible [25].

To make the GPS spoofing attacks fail, Global Navigation Satellite System (GNSS) Receiver Stand-alone (GRS) [26], and Hybrid Position Receiver (HPR) [27] methods are commonly employed. GRS techniques rely on spatial processing [28–30] and residual signal defense [31]. Further elaboration on these methodologies can be found in Spanghero and Papadimitratos [32], Pullen and Joerger [33]. Additionally, drawbacks of HPR techniques are detailed in Mahmood and Manivannan [34], Gholami et al. [35]. Various components such as control systems, actuators, sensors, and guidance, mounted on Drones are inherently vulnerable to spoofing attacks [36]. In such cases, the integrity of the mission can be compromised by the injection of fake data, such as incorrect landing trajectories or erroneous target locations [37]. Employing inexpensive hardware or installing weak software makes Drones susceptible to hacking through the injection of fake data, and spoofing attacks. The spoofing attacks can manifest in different forms [38], as detailed below:

- **Basic Spoofing:** In this type of attack, eavesdroppers generate counterfeit Global GNSS signals, which can be executed using inexpensive software and hardware.
- **Intermediate Spoofing:** The attackers produce fake signals while simultaneously targeting each channel of the victim receiver by aligning the code phases between legitimate and incoming spoofed signals.
- **Spoofing with Multiple Antennas:** This technique primarily employed against receivers with multiple antennas, where attackers generate multiple signals to disrupt the frequencies of other signals.

To protect Drones against GPS spoofing attacks, a ML based model proposed in this research. Upon reception of a GPS signal by the Drone, the proposed system for detecting spoofed signals is activated. The Drone remains inactive until it receives a GPS signal from an authenticated source. When a counterfeit GPS signal be detected, the Drone will wait for the subsequent signal until an authenticated one is received.

The details of the features used to classify counterfeit and authentic GPS signals in this research are explained in Section 4.4. Upon the reception of an authenticated GPS signal, the UAV will proceed with the appropriate action. The primary advantages of the proposed model lie in its computational efficiency and suitability for real-time applications, ensuring ease of implementation.

1.1. Organization of the paper

The remaining sections of the paper are structured as follows: Section 2 presents an overview of existing methodologies, along with vulnerabilities and their potential solutions. Section 3 delves into preliminary knowledge, which covers foundational aspects essential for building the proposed machine learning model. Section 4 provides detailed insights into the proposed system design. In Section 5, the experimentation and evaluation of the proposed model are presented. Moreover, a comparative analysis with existing models is also made. Finally, Section 7 conclude the proposed work and provide future recommendations, respectively.

2. Related work

This section serves to explore and evaluate the current state-of-the-art methodologies employed in enhancing the robustness of the autonomous systems against GPS spoofing threats. By examining prior research methodologies, this section aims to provide valuable insights into the evolving landscape of security challenges faced by autonomous systems. In recent years, numerous approaches have been proposed to detect spoofing attacks and protect autonomous systems. Some of these approaches have demonstrated promising accuracy, i.e., more than 90%. However, a significant gap remains. The gap refers to the limitations in existing methodologies that, despite achieving high accuracy rates (over 90%), are still not sufficient to protect autonomous systems against GPS spoofing threats. This gap could involve a number of factors, such as the adaptability of spoofing techniques, the robustness of detection methods, and real-time processing capabilities. Addressing this gap is crucial to ensure robust protection of autonomous systems against sophisticated cyber threats.

In Dang et al. [39], Dang et al. presented an approach that utilizes deep ensemble learning methods in a mobile-network-assisted UAV monitoring system to detect GPS spoofing in cellular-connected UAVs. The method uses path losses between base stations (BSs) and UAVs to detect deviations caused by spoofing by employing three statistical methods to enhance accuracy. Deep ensemble learning on edge cloud servers analyses path loss statistics using multilayer perceptron (MLP) neural networks to make decisions without adding energy consumption to UAVs [40]. However, reliance on path losses

makes the system suspectable to environmental distortion. Further, by deploying deep learning on edge cloud servers raises challenges about data integrity and privacy. In Umer et al. [41], Umer et al. introduced a stacked ensemble method for detecting GPS signal spoofing in small UAVs. The authors constructed a model using support vector machine [42] and convolutional neural networks which is validated through controlled simulation tests and compared against traditional machine learning approaches. Despite achieving a remarkable accuracy rate of 99.74% a few challenges exist due to the reliance on simulated tests and potential computational complexity that highlights issues real-world deployment. In Nayfeh et al. [43], Nayyfeh et al. developed an ML model to detect and classify GPS spoofing in UAVs that is validated through controlled outdoor scenarios covering static and dynamic attacks. Authentic GPS signal data and spoofed sets generated with SDR transceiver modules are standardized and analyzed before training multiple ML classifiers. Limitations of their proposed work includes non-robustness against quantum cyber attacks. Moreover, the challenges might occur due to potential discrepancies between controlled setups and real-world conditions. In Agyapong et al. [44], Agyapong et al. developed deep learning models utilizing UAV flight logs and telemetry data to detect GPS spoofing attacks in real-time. These models, trained on generated UAV data across various models such as a Long Short-Term Memory (LSTM) binary classifier and an LSTM autoencoder-based one-class classifier (OCC), Efficiency is evaluated through simulation tests and hardware validation experiments, showcasing high accuracies. However, reliance on simulated tests and uncertainty regarding real-world efficacy, particularly against spoofing techniques creates notable vulnerabilities to practical deployment. In Pawlak et al. [45], Pawlak et al. introduced a machine learning framework for detecting and classifying jamming attacks on UAVs. Qualitative assessments of attack impact on drones utilizing orthogonal frequency-division multiplexing (OFDM) communication are conducted, with radiometric parameters recorded before and after each attack. Six ML algorithms utilized features extracted from these parameters to autonomously detect and classify jamming attacks and achieved an accuracy of 92.2% with a false alarm rate of 1.35%. However, the reliance on systematic testing scenarios and software-defined radio (SDR) implementations can limit generalizability, while the effectiveness of the ML algorithms against sophisticated adversaries warrants further validation in complex threat landscapes.

In Iqbal et al. [46], Iqbal et al. investigated advanced methods to enhance GNSS security against spoofing attacks. Traditional supervised machine learning requires extensive training data for all potential attacks but fails with new vectors. To address this, the authors used representation learning-based methods. Their classification classify new samples based on single-class training data. They also introduce a GNSS spoof detection model combining a Variational AutoEncoder (VAE) and a Generative Adversarial Network (GAN). This is trained on features from standard GNSS receivers using the Texas Spoofing Test Battery (TEXBAT) datasets. Their model is susceptible to spoofing attacks that closely mimic legitimate signals and heavily depends on comprehensive training data. In Borhani-Darian et al. [47], Borhani et al. proposed a deep learning-based algorithm for detecting spoofing attacks. Their model utilizes a parallelized deep learning model and a clustering algorithm to estimate spoofing signals. Their model misclassify spoofing signals that closely resemble legitimate ones, and its effectiveness could diminish in complex environments because of its dependence on accurate clustering. In Abrar et al. [48], Abrar et al. proposed an integration of a GPS navigation model with a dynamic bicycle model to represent the behavior of autonomous vehicles. The algorithm utilized machine learning to assess temporal features and distinguish between typical and anomalous navigation behaviors. Nonetheless, the vulnerability of the system is against the advanced spoofing attacks that imitate regular vehicle behavior, which ultimately lead to inaccurate results.

In Dang et al. [49], Dang et al. introduced a cellular network-assisted system to monitor UAV positions and combat GPS spoofing using deep learning. Their approach utilizes an MLP model trained on path loss measurements from nearby base stations to authenticate GPS positions. While achieving high detection accuracies in experimental tests, a few vulnerabilities arise from signal interference and environmental factors that affects path loss measurements. Moreover, uncertainties persist regarding the efficacy of the deep learning approach against advanced GPS spoofing techniques, requiring further validation against evolving threats. In Tlili et al. [50], Tlili et al. introduced a novel hybrid adaptive framework for fault and attack detection in UAVs. By employing two entry flows to learn high-level features from data and utilizing deep-learning architectures for validation, the framework demonstrated the accuracies of 85% for UAV faults and 96.7% for attacks. However, vulnerabilities arise from reliance on deep-learning architectures that creates challenges related to data quality and model generalization. In Brewington and Kar [51], Brewington et al. examined neural generative one-class classifiers such as Generative Adversarial Networks (GANs) for GPS spoofing detection in civilian UAVs. The vulnerabilities arise from the adversarial attacks and data manipulation which creates risks to detection integrity. In Wei et al. [52], Wei et al. introduced SigFeaDet, a machine learning-based approach for detecting GNSS spoofing in UAVs. Through employing various machine learning algorithms trained on GNSS signal data, the method achieves a detection accuracy of 95% with an equal error rate (EER) of approximately 5%. The weakness of their proposed work arises arise from signal variations due to environmental factors or hardware malfunctions which ultimately leads to inaccuracies in detection accuracy. Moreover, the uncertainties exists regarding SigFeaDet's robustness against sophisticated spoofing techniques which need further validation and deep experimentation with highly robust systems. In Aladi and Alsusa [53], Aladi et al. introduced a method that utilizes physical layer features in UAV systems to enhance security. Using received signal strength (RSS) and angle of arrival (AoA), they developed a localization-based verification system for signal authentication by employing a deep learning autoencoder. Despite achieving high accuracy, vulnerabilities of their proposed work are related to the signal variations due to environmental conditions or hardware inconsistencies. The summaries of the existing work is provided in Table 1.

2.1. Contributions of the paper

To overcome the vulnerabilities mentioned in Table 1 and detect whether the signal is spoofed or authentic, the following contributions are made in this work:

- The proposed research presents an integrated machine learning methodology designed to intelligently detect GPS spoofing attacks aimed at compromising the security of autonomous systems, particularly autonomous systems.
- The methodology utilizes signal features associated with mathematical concepts such as jitter, shimmer, and frequency modulations to effectively detect spoofing attacks injected into GPS signals.
- The process of collecting and segmenting a private dataset based on statistical signal features into intervals specific to spoofing and authentic signals demonstrates a tailored approach to dataset preparation, which enhances the accuracy and reliability of the proposed model.
- Leveraging the RSS and data obtained from legitimate LoRaWAN devices, a single-class support vector machine (SCSVM) classifier is developed for detecting spoofing attacks.

The adoption of shimmer and jitter in this work is motivated by their prior use in speech and biomedical signal analysis for identifying subtle variations in timing and amplitude [54,55]. Our contribution is not the introduction of shimmer and jitter as new features, but their application for GPS spoofing detection in UAVs. This cross-domain adaptation applies established signal processing concepts to enhance UAV security.

Table 1
Summary of the existing work.

Methodology	Application domain	Real-world performance	Robustness against attacks	Advantages	Disadvantages	Potential solutions
Dang et al. [39]	Mobile UAV monitoring	Uses path losses	Affected by environment	Energy-efficient, accurate	Data integrity, privacy issues	Improve robustness, encryption
Umer et al. [41]	Small UAVs	99.74% accuracy	Simulated tests only	High accuracy, ensemble model	High computation, deployment issues	Add real tests, optimize
Nayfeh et al. [43]	UAVs	Outdoor scenario validation	Weak to quantum attacks	Standardized data analysis	Controlled vs real mismatch	Quantum-safe algorithms, validate
Agyapong et al. [44]	Real-time monitoring	Simulated and hardware validation	Simulated test reliance	Uses flight telemetry logs	Unknown spoofing resilience	Real tests, adaptive learning
Pawlak et al. [45]	UAVs	92.2% accuracy	Systematic test reliance	Detects jamming attacks	Low generalizability	Diverse tests, robust ML
Dang et al. [49]	UAV position monitoring	High experimental accuracy	Sensitive to interference	MLP on path loss	Weak against spoofing	Reduce interference, enhance models
Tlili et al. [50]	Fault/attack detection	85%/96.7% accuracy	Deep learning reliance	Hybrid adaptive method	Data, model generalization	Better data, interpretability
Brewington et al. [51]	Civilian UAVs	Adversarial vulnerabilities	Detection integrity risks	Neural one-class classifiers	Not specified	Harden algorithms, validate data
Wei et al. [52]	UAVs	95% detection, 5% EER	Environmental signal issues	Various ML algorithms	Poor spoofing resilience	Robust signals, broad tests
Aladi et al. [53]	UAV systems	High accuracy via physical layer	Environmental signal variance	Uses RSS, AoA	Signal variation vulnerability	Improve signal authentication

3. Preliminaries

To develop a proposed model for detecting the nature of the received signals, whether they are authentic or spoofed, through the utilization of machine learning algorithms, the preliminary knowledge used is explained in the next subsections.

3.1. Two-way technology communication

Two-way technology communication involves the transmission of signals between different wireless technologies [56,57]. In contrast to the modulation based on the packet-level, physical-level modulation offers finer control which helps to enable fast-speed throughput by manipulating diverse signals at the physical layer. In a ground breaking research, (Long Range Wide Area Network (LoRaWAN) [58] utilizes a WiFi signal to replicate a LoRaWAN signal without requiring changes to hardware or firmware. The LoRaWAN meticulously designs the content of a transmitted WiFi communication packet to resemble the radio frequency (RF) waveform characteristics of LoRaWAN signals. Upon receiving such a WiFi frame, LoRaWAN devices dismiss the WiFi header, trailer, and preamble as extraneous noise [59]. However, the payload effectively navigates through LoRaWAN's preamble detection process, which enables the LoRaWAN receiver to decode the emulated LoRaWAN frame.

In this research, LoRa and LoRaWAN are used with their distinct technical meanings. LoRa refers to the proprietary chirp spread spectrum (CSS) modulation at the physical layer, responsible for long-range, low-power transmission. LoRaWAN, in contrast, is the medium-access and network-level protocol that manages device authentication, packet routing, and data integrity across gateways.

3.2. Theoretical examination of spatial correlation

The theoretical examination of spatial correlation is performed by analyzing how measurements (like RSS) taken at different locations are statistically related. The RSS analyzed in the proposed research is closely tied to the physical positioning within a given space and is readily accessible within existing wireless networks. Despite being subject to random noise, environmental fluctuations, and multipath effects, RSS readings taken from the same physical location exhibit similarity, while those from different spatial positions display noticeable differences. Consequently, RSS values demonstrate robust spatial correlation features [60–62].

The RSS value vector is defined as $V = \{V_1, V_2, \ldots, V_N\}$, where N represents the total number of reference points. Such reference points establish their locations by collecting RSS statistical values from wireless junctions. Typically, the RSS statistical value of a wireless junction obtained at the ith reference point adheres to the following Log-scale distribution as given in Eq. (1) [63]:

$$V_j(d_i)[dC_n] = P(d_0)[dC_n] - 10\alpha log\left(\frac{d_i}{d_0}\right) + Y_j \quad (1)$$

Where $P(d_0)$ shows the maximum power that can be provided by the sensors at the reference range d_0, d_i denotes the length from sensor i to the jth reference point, α indicates the exponent of the path loss, and Y_j exhibits shadow fading, characterized by a Zero-Mean Gaussian Distribution (ZMGD) [64] with a Standard Deviation (SD) [65,66] of δ. For simplicity, it is assumed that the transmission power of all wireless devices is the same. The RSS distance from one device to another in the signal space of the jth reference point can be expressed as mentioned in Eq. (2):

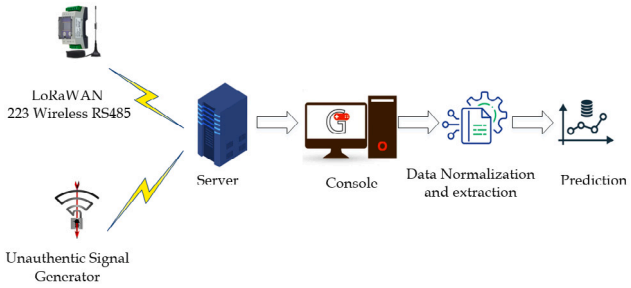


Fig. 2. Generalized spoof signal detection flow.

$$\Delta V_j = 10\alpha log\left(\frac{d_n}{d_{n-1}}\right) + \Delta Y \tag{2}$$

Where ΔY follows a ZMGD with a δ of $\sqrt{2\delta}$. The squared value of RSS distance in N reference points as given in Eq. (3):

$$\Delta D^2 = \sum_{j=1}^{N} \Delta V_j^2 \tag{3}$$

Where ΔV_j with $j = 1, 2, \ldots, N$ denotes the RSS distance at the jth reference point, represented by Eq. (2).

4. System design

The details of the proposed system designed for the detection of spoof or authentic signals are given in the next few subsections.

4.1. Network architecture

The network architecture, depicted in Fig. 2, comprises wireless devices (including LoRaWAN and WiFi devices), a server, and a console. In a scenario where regular communication between LoRaWAN devices is ongoing, an adversary could exploit the direct communication capability between WiFi and LoRaWAN devices to execute a spoofing attack. In such instances, monitoring sensors positioned at stationary locations can receive WiFi eavesdropper frames in real time to detect spoofing actions. The server then processes the packets supervised by these sensors for overarching detection, while the console receives data packets, normalizes RSS samples using timestamps or sequence numbers, merges the data packets, and develops the samples.

Fig. 3 outlines the training process for the Single-class Support Vector Machine (SCSVM) model. Before employing the SCSVM model for the classification of RSS samples in real-time scenarios (in the presence of spoofing attacks), a collection of authentic RSS samples must be used to train the model. Initially, all incoming RSS samples of the data undergo preprocessing, which includes the following steps: (1) Eliminating outliers by applying the 3σ criterion to filter noise data; and (2) Filling missing values using the K-nearest Neighbors (K-NN) data filling algorithm to efficiently handle continuous noisy data, and missing values. The preprocessed data is then split into a couple of parts: (a) the training part, and (b) the testing part, ensuring the full applicability of the SCSVM model.

Fig. 4 outlines the flow of the detection of spoofing attacks. Once the model is developed, the detection process of spoofing is initiated. Finally, the SCSVM classifier is employed to categorize incoming RSS data into spoofing and authentic data.

4.2. GNSS data collection and integration

The primary objective of this research is to create a controlled environment to systematically study and evaluate the effectiveness of our proposed machine learning-based approach for detecting GPS spoofing attacks. In this context, LoRaWAN signals are utilized to emulate certain aspects of GPS signal interference because they provide a practical and flexible way to model signal anomalies in a controlled manner. However, using true GNSS data provides a more realistic and representative evaluation of the proposed model. Therefore, genuine GNSS data is collected and integrated with the existing dataset that contains LoRaWAN signals.

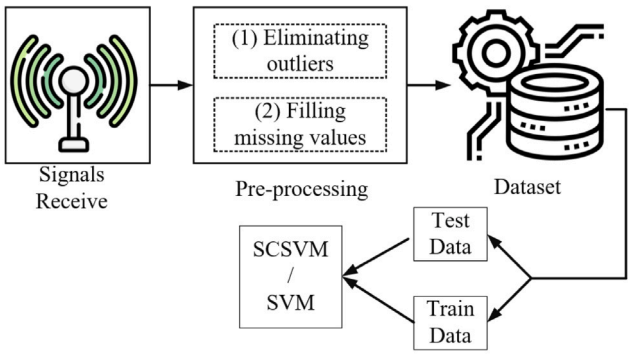


Fig. 3. Train the SCSVM/SVM model.

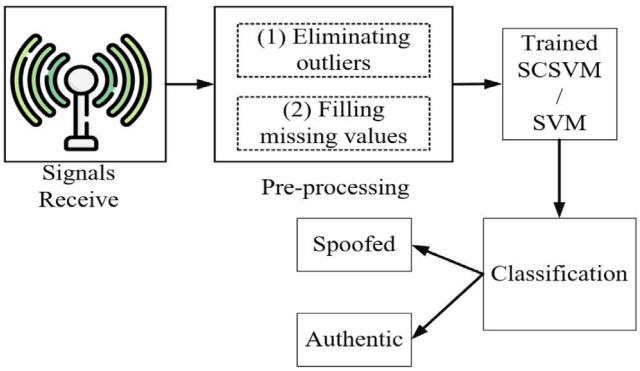


Fig. 4. Spoofing detection flow.

4.2.1. Data collection process

To capture the signals from GPS satellites, a high-precision GNSS receiver such as Real-Time Kinematic (RKT) [67] is utilized. This receiver decodes signals transmitted by satellites and provides raw data, including the satellite position, signal strength, and timing information. After that, it is ensured that the receiver can log raw signal data, which can later be processed and analyzed. Eq. (4) shows the pseudorange calculation to estimate the distance between a satellite and a GNSS receiver. The term "pseudo-range" is used because this distance measurement includes not only the true range between the satellite and the receiver but also several sources of error, such as clock biases, atmospheric delays, and multipath effects.

$$\rho_i = (t_{rcv} - t_{trans,i}) \cdot c \tag{4}$$

where ρ_i is the pseudo-range to the ith satellite, t_{rcv} is the receiver's clock time when the signal is received, $t_{trans,i}$ is the satellite's clock time when the signal was transmitted, and c is the speed of light.

To calculate the position, let (x_i, y_i, z_i) be the coordinates of the ith satellite, (x, y, z) be the unknown receiver coordinates, and $c \cdot \delta t$ be the clock bias. Eq. (5) is used to determine the position of a receiver on Earth, such as a GPS device.

$$\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2} = \rho_i + c \cdot \delta t \tag{5}$$

The signal strength (S) is calculated using Eq. (6).

$$S(d) = \frac{S_0}{d^2} + \eta \tag{6}$$

where S_0 is the signal strength at a reference distance, d is the distance from the signal source, and η represents random fluctuations due to weather and other environmental factors.

To improve the accuracy and reliability of GNSS positioning, atmospheric effects, specifically ionospheric and tropospheric delays are calculated using Eq. (7).

$$\begin{cases} \Delta t_{ion} = \frac{40.3}{f^2} \cdot TEC \\ \Delta t_{trop} = \frac{P}{c \cdot T} \cdot (k_1 + k_2 \cdot \frac{e}{T}) \end{cases} \tag{7}$$

where f is the frequency of the signal, TEC is the total electron content, P is atmospheric pressure, T is temperature, e is partial water vapor pressure, and k_1 and k_2 are constants.

The carrier phase is measured using $\Phi = N + \phi$ (Φ is the observed phase, N is the integer ambiguity) for high-precision applications like surveying and real-time kinematic GNSS due to its ability to provide much finer measurements than pseudorange alone.

Now, the Doppler shift is measured after carrier phase measurement to determine the relative velocity between the GNSS receiver and the satellite using Eq. (8). It provides additional information on the rate of change of distance, which helps in improving position accuracy and estimating the receiver's velocity

$$f_D = \frac{v_r \cdot f_0}{c} \tag{8}$$

where f_D is the Doppler shift, v_r is the relative velocity between the satellite and receiver, f_0 is the transmitted frequency, and c is the speed of light.

After, the measurement of Doppler shift, error correction (e.g., Multipath, Atmospheric) is calculated using the correction model like the Klobuchar model for ionospheric delay [68] using Eq. (9)

$$\Delta t_{ion} = F \cdot (a_0 + a_1 \cdot \psi + a_2 \cdot \psi^2 + a_3 \cdot \psi^3) \tag{9}$$

where F is the obliquity factor, and ψ is the zenith angle. The least squares estimation for positioning is measured as: $\mathbf{x} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$. $\mathbf{H}$ is the design matrix, $\mathbf{y}$ is the vector of observations, and $\mathbf{x}$ is the estimated position. In the last step, combine GNSS and other datasets (e.g., LoRaWAN) using Kalman filtering due to its ability to optimally combine predictions with noisy measurements which also provides accurate state estimates even in the presence of uncertainty and dynamic system behavior. Mathematically, the Kalman filtering is described in Eq. (10).

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \tag{10}$$

where $\hat{\mathbf{x}}_{k|k}$ is the state estimate, $\mathbf{K}_k$ is the Kalman gain, and $\mathbf{z}_k$ is the measurement.

4.2.2. Signal segmentation approach

To prepare the captured GPS/LoRaWAN signals for feature extraction and spoofing detection, the raw time-series data are segmented into overlapping temporal windows. This process provides the model to isolate short-term variations caused by spoofing while preserving continuity in authentic signal behavior.

- Step 1: Signal representation: Let the received signal be denoted as: $s(t) = A(t)\cos(2\pi f_c t + \phi(t))$, where $A(t)$ is the instantaneous amplitude, f_c is the carrier frequency, and $\phi(t)$ is the phase component, which may contain small perturbations due to spoofing (timing offsets, phase jumps).
  The discrete sampled signal can be represented as: $s[n], \quad n = 1, 2, \ldots, N$, where N is the total number of samples in the recording.
- **Step 2: Window segmentation:** Each signal trace is divided into fixed-length windows of L samples (corresponding to 200 ms). Consecutive windows overlap by 50%, giving a stride of $\frac{L}{2}$. Formally, the ith window is defined as: $w_i[n] = s[n+(i-1)\cdot\frac{L}{2}], \quad n = 1, \ldots, L$. Hence, the total number of windows K generated from a signal of N samples is: $K = \left\lfloor \frac{2N}{L} - 1 \right\rfloor$ This ensures that each new segment reuses half of the previous samples, maintaining temporal continuity and avoiding information loss at segment boundaries.
- **Step 3: Feature extraction per segment:** For each window ($w_i[n]$), first, Compute jitter, shimmer, and frequency-modulation parameters using equations already defined in Table 3. Second, estimate RSS using Eq. (11)

$$\mathrm{RSS}_i = 10\log_{10}\left(\frac{1}{L}\sum_{n=1}^{L}|w_i[n]|^2\right) \tag{11}$$

  Third, form a feature vector: $\mathbf{x}_i =, \mathrm{RSS}_i, J_i, S_i, \mathrm{FM}_i$, where ($J_i$) and ($S_i$) are the jitter and shimmer of segment i.
- **Step 4: Label assignment:** Each window inherits the label of its parent signal trace as given in Eq. (12).

$$y_i = \begin{cases} 1, & \text{if segment from authentic transmission} \\ -1, & \text{if segment from spoofed transmission} \end{cases} \tag{12}$$

  Labeling is based on the controlled experimental setup: Authentic signals → LoRaWAN transmitters, and Spoofed signals → HackRF. One re-transmissions with timing and amplitude manipulation.
- **Step 5: Dataset formation:** The final dataset used for model training is: $D = (\mathbf{x}_i, y_i)_{i=1}^{K}$. This segmented dataset is normalized using z-score normalization and then split into training (70%) and testing (30%) subsets.

The selection of a 200 ms window is consistent with previous findings [69,70], which report that window durations of 100–250 ms provide sufficient temporal resolution to capture rapid signal perturbations associated with spoofing. A 200 ms window provides a balanced configuration by offering enough samples to estimate shimmer and jitter reliably, maintaining adequate temporal resolution to detect rapid signal anomalies, and ensuring manageable computational complexity during processing.

4.3. Dataset composition and structure

The dataset used in this research consists of 6000 signal samples collected over multiple experimental sessions using GPS and LoRaWAN receivers. Data were recorded under diverse conditions, including daylight, low-light, and varying device distances (3–8 m), to ensure heterogeneity. Each sample contains time-synchronized raw signal attributes such as received signal strength (RSS), amplitude, frequency modulation parameters, and extracted jitter and shimmer values. The dataset is divided into two subsets: 70% for training and 30% for testing. Signal labeling was performed manually based on controlled spoofing experiments using HackRF One devices. Signals received directly from legitimate LoRaWAN transmitters were labeled as authentic, whereas those intercepted and retransmitted with altered identifiers or timing offsets were labeled as spoofed. This labeling approach ensures that spoofed samples accurately reflect realistic attack behavior while maintaining a clear distinction from genuine signals. Table 2 outlines the key parameters, acquisition conditions, and labeling procedure used during data collection.

4.4. Features used in the proposed work

In the proposed research, we utilize signal attributes such as "shimmer" and "jitter", along with their subcategories, as key features. These

Table 2
Dataset summary and structure used in this research.

Category	Description
Total Samples	6000 signal instances collected over multiple controlled sessions.
Signal Sources	Mixed GPS and LoRaWAN transmissions captured using Pycom LoPy4 nodes (authentic) and HackRF One devices (spoofed).
Environment Settings	Indoor laboratory setup with variable lighting (daylight/low-light) and distances ranging from 3 m to 8 m between transmitter and receiver
Sessions Conducted	10 independent sessions (5 daylight, 5 low-light) conducted over different days to capture environmental variability
Signal Features Recorded	Received Signal Strength (RSS), Jitter, Shimmer, Frequency Modulation (FM), amplitude, and timestamp metadata
Data Structure	Each sample stored as a vector: [RSS, Jitter, Shimmer, FM, TimeStamp, Label]. Average record size ≈ 1.2 KB
Labeling Criteria	• Authentic signals: Directly received from legitimate LoRaWAN transmitters without alteration • Spoofed signals: Captured via HackRF One re-transmission with modified timing, amplitude, or identifiers to emulate GPS spoofing attacks
Class Distribution	3000 Authentic (50%) and 3000 Spoofed (50%)
Training/Testing Split	70% training (4,200 samples) and 30% testing (1,800 samples)
Sampling Rate	2 MHz for GNSS/GPS; 868 MHz band for LoRaWAN signals
Collection Duration	Approximately 30 min per session (total ≈ 5 h of data)
Signal Preprocessing	Outlier removal via 3σ criterion, K-NN interpolation for missing values, and z-score normalization prior to feature extraction
Feature Extraction Windows	200 ms sliding windows with 50% overlap applied to all signal traces

features are used to discern the authenticity of a received signal. Upon receiving a signal, our algorithm extracts feature values from it, creating a feature vector denoted as $T_{vec} = t_1, t_2, t_3, t_4,.....t_n$. Subsequently, these values are compared with each feature vector in the dataset using our proposed model. If the model determines that the signal originates from an authentic source, the UAV will be poised to take appropriate action.

Conversely, if the signal is deemed fake or spoofed, the UAV will abstain from taking any subsequent action and will await additional incoming signals. This cycle of signal reception will continue until the UAV receives an authentic signal. However, rather than remaining completely stand-still, the UAV is also programmed to execute a predefined safe behavior during this period, such as hovering in place, returning to a known safe location, or maintaining its last known trajectory at a reduced speed. This ensures that the UAV does not remain vulnerable or inactive but instead takes precautionary measures until a legitimate signal is verified. This approach balances security with operational continuity, minimizing the risk posed by spoofed signals while maintaining a level of responsiveness. Upon acquiring an authentic signal, the UAV will proceed with the appropriate action. After completion of the task, the UAV will iterate through the same process for subsequent signals.

Fig. 5 illustrates a simplified scenario where the UAV is shown receiving control signals. However, the UAV in the proposed model also relies on GNSS signals for navigation and localization. Fig. 5 also emphasizes the decision-making process related to signal authenticity. While the UAV receives navigation data from GNSS, it also communicates with the ground station for additional commands or mission updates. The proposed model is designed to assess the authenticity of these incoming signals — whether from GNSS or control station — to ensure the UAV only acts on verified information.

In the signal classification approach, we employ jitter and Shimmer as primary features. Although previous studies [71–73] have explored additional signal attributes such as mean, frequency, range, and root mean square, they have found that relying only on these features results in inadequate accuracy for signal classification. Consequently, to enhance the accuracy of our model, there are ten parameters as features are incorporated, as opposed to just two or three. The parameters employed are elaborated in the Table provided in 3, wherein mathematical specifications of these features such as the intervals defined for each

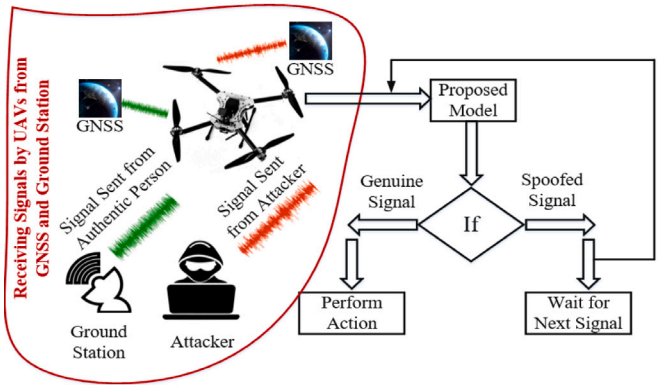


Fig. 5. UAV response to signal authenticity.

classification, are outlined. However, the detailed explanation of these mathematical equations, as presented in Table 3, can be found in Silva et al. [74], Teixeira et al. [75], Teixeira et al. [76].

Jitter and Shimmer are vital for detecting spoofing attacks in UAVs because they highlight inconsistencies in signal timing and amplitude, respectively. Normally, UAV signals exhibit minimal jitter and a stable shimmer that indicates consistent timing and amplitude. However, spoofing attempts introduce irregularities and randomness, which cause increased jitter (small, rapid timing variations) and shimmer (amplitude variability).

In our approach to detecting GPS spoofing attacks, we utilize jitter, shimmer, and modulation techniques to analyze signal anomalies. Jitter, which refers to variations in the timing of signal pulses, helps identify irregularities that may indicate spoofing by highlighting deviations from expected timing patterns. Moreover, Shimmer is used to measure the amplitude variation, and it is initially considered to capture subtle deviations in signal strength, but it is recognized in this research that amplitude variation as a general concept can offer a more robust approach. Therefore, in this research, shimmer is used to detect amplitude variations in signal strength. Additionally, modulation is used to examine the signal's encoding patterns, as spoofed signals often exhibit irregular modulation compared to authentic ones. By integrating these mathematical concepts as given in Table 3, the proposed

Table 3
Detail of features used in the proposed work.

Features	Mathematical equations	Variable explanation	Defined intervals
Jitter (absolute)	$= \frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert T_i - T_{i-1}\rvert)$	T_i: time span of each period in N: total number of periods	[01.00 25.50] ⇒ Spoofed signal [25.75 75.55] ⇒ Authentic signal [75.50 100.0] ⇒ Spoofed signal
Jitter (local)	$= \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert T_i - T_{i-1}\rvert)}{\frac{1}{N}-\sum_{i=1}^{N}(T_i)}$	//	[01.00 25.50] ⇒ Spoofed signal [25.75 75.50] ⇒ Authentic signal [75.75 100.0] ⇒ Spoofed signal
Jitter (RAP)	$= \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert T_i - (\frac{1}{3}\sum_{n=i-1}^{i+1}(T_n))\rvert)}{\frac{1}{N}-\sum_{i=1}^{N}(T_i)}$	//	[100.0 75.50] ⇒ Spoofed signal [75.00 25.50] ⇒ Authentic signal [25.25 01.00] ⇒ Spoofed signal
Jitter (ppq5)	$= \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert T_i - (\frac{1}{5}\sum_{n=i-2}^{i+2}(T_n))\rvert)}{\frac{1}{N}-\sum_{j=1}^{N}(T_i)}$	//	[01.00 25.50] ⇒ Spoofed signal [25.75 75.50] ⇒ Authentic signal [75.50 100.0] ⇒ Spoofed signal
Shimmer (local)	$= \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert A_i - A_{i+1}\rvert)}{\frac{1}{N}-\sum_{i=1}^{N}(A_i)} \times 100$	A_i: Amplitude of the received signal	[100.0 75.50] ⇒ Spoofed signal [75.00 25.50] ⇒ Authentic signal [25.55 01.00] ⇒ Spoofed signal
Shimmer(db)	$= \frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert 20 * log(\frac{A_{i=i+1}}{A_i})\rvert)$	//	[00.50 24.75] ⇒ Spoofed signal [25.00 74.75] ⇒ Authentic signal [75.00 99.50] ⇒ Spoofed signal
Shimmer (apq3)	$= \frac{\frac{1}{N-1}\sum_{i=1}^{N-1}(\lvert A_i - (\frac{1}{3}\sum_{n=i-1}^{i+1}(A_n))\rvert)}{\frac{1}{N}-\sum_{i=1}^{N}(A_i)}$	//	[100.0 75.50] ⇒ Spoofed signal [75.00 25.50] ⇒ Authenticsignal [25.25 01.00] ⇒ Spoofed signal
Shimmer (apq5)	$= \frac{\frac{1}{N-1}\sum_{i=2}^{N-2}(\lvert A_i - (\frac{1}{5}\sum_{n=i-2}^{i+2}(A_n))\rvert)}{\frac{1}{N}-\sum_{i=1}^{N}(A_i)}$	//	[01.00 25.50] ⇒ Spoofed signal [25.75 75.25] ⇒ Authentic signal [75.5 100] ⇒ Spoofed signal

approach aims to enhance the detection of spoofing attempts through a comprehensive analysis of signal characteristics. Moreover, Using shimmer in the proposed model adds a layer of complexity for attackers because shimmer captures subtle and rapid variations in the amplitude of a signal, which are typically caused by the physical characteristics of the legitimate signal's source. These variations are difficult to replicate accurately because they are influenced by a combination of factors such as the environment, the transmission medium, and the intrinsic properties of the legitimate transmitter. An attacker attempting to spoof the signal would need to precisely mimic these nuanced amplitude fluctuations in real-time, which is technically challenging and requires sophisticated equipment and knowledge. This makes shimmer a valuable feature in detecting spoofed signals, as it introduces a unique and dynamic characteristic that is hard to counterfeit.

Recent research [77–79] also showed that spoofed signals can cause jitter to increase by over 20% and shimmer to rise by more than 15%, making these features reliable indicators of potential spoofing attacks. Spoof signal detection is based on signal processing principles, which predict normal and spoofed signal behavior.

Apart from the features mentioned in Table 3, another feature used in this research is frequency modulation, which involves the blending of an authentic signal with a high-frequency or carrier signal. Despite the inherent high frequency of GPS signals, security measures mandate their modulation with a specific frequency range, typically between 250 MHz and 300 MHz, prior to transmission. The demodulator installed on the UAV is designed to work specifically with the signal structure expected from the GNSS system. It is tuned to identify whether the incoming signal conforms to the expected GPS signal characteristics, particularly within the designated frequency range. Any signal, including those modulated with LoRa techniques, that falls outside this expected range or does not match the anticipated GNSS signal parameters, would be flagged as suspicious. The distinction lies in the purpose of the demodulation process in our model. It is not intended to decode LoRa signals but rather to ensure the integrity of GNSS signals by identifying anomalies. Therefore, while LoRa's signal structure is indeed different, our model is robust enough to distinguish between legitimate GPS signals and potential spoofed signals, even if the latter employs different modulation techniques like those used in LoRa.

Potential attackers can attempt to send GPS signals through two methods: (a) modulated GPS signals and (b) unmodulated GPS signals. However, since the carrier signal's frequency, which is used for modulation, remains undisclosed to attackers, Therefore, it serves as an additional secret key, further enhancing system security. The generalized flow of the proposed mode is presented in Fig. 6.

Shimmer and jitter are included in this research based on their established utility in earlier research across non-GPS domains [55,80]. In this research, these features are specifically adapted for GPS spoofing detection, where spoofed signals cause irregularities in timing (captured by jitter) and amplitude (captured by shimmer). This targeted application differentiates the proposed work from earlier studies in the UAV security context.

4.5. Detection of attacks incorporating SCSVM analysis

An instance to elucidate the signal spoofing attack executed by a WiFi device opposed to LoRaWAN devices. Due to the inherent transmitting nature of the wireless medium, WiFi devices situated in close proximity can readily intercept data chunks transmitted by LoRaWAN transmitters. Consequently, the WiFi evesdropper initiates a signal spoofing attack involving a couple of steps.

Initially, the attacker enters a state of channel listening, capturing the transmission frame from the LoRaWAN device, then alters the identity information to impersonate an authentic LoRaWAN device, and finally launches a unauthorized attack. Detection of the spoofing attack involves monitoring the RSS value of the WiFi data packets employing the monitoring sensors. This research analyzes RSS values from WiFi devices gathered by monitoring sensors placed in different locations to monitor potential attackers. The RSS value of the data packet transmitted by wireless device j is represented by Eq. (13).

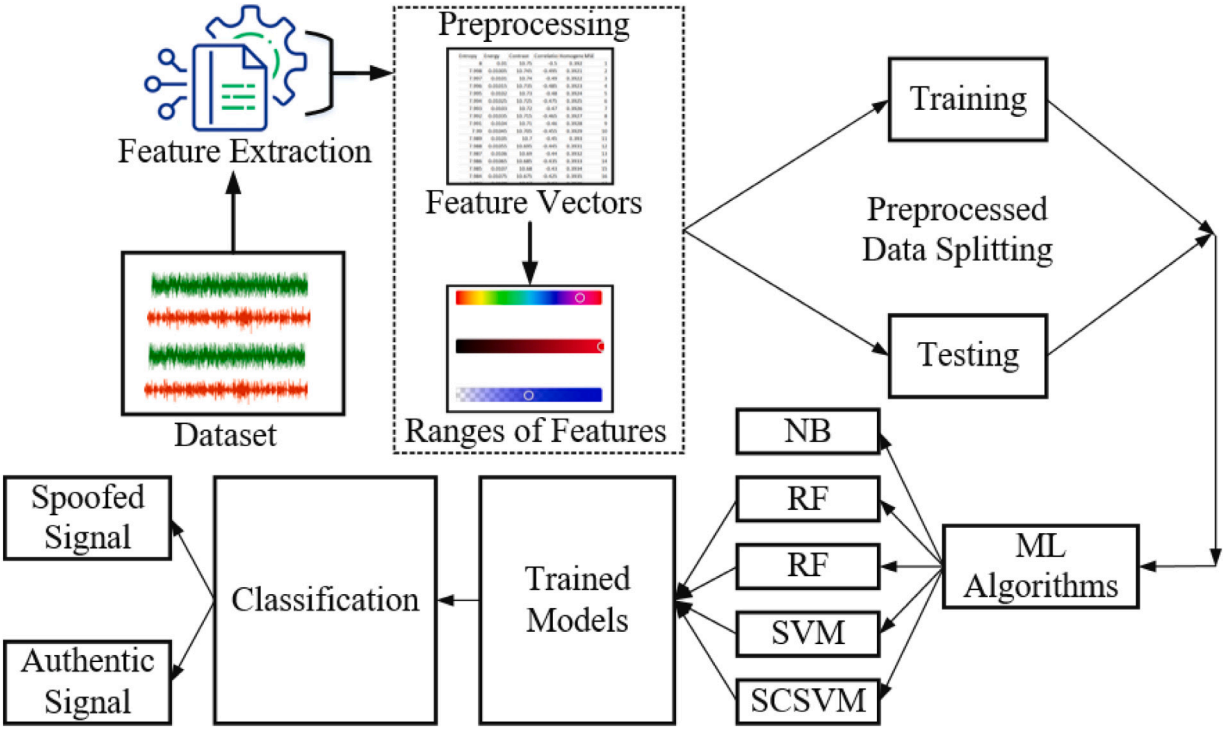


Fig. 6. Generalized flow diagram of the proposed model.

$$y_j = \left[RSS_j^1, RSS_j^2, RSS_j^L, \ldots\right] \tag{13}$$

Where, RSS_j^L is the RSS value of node at j from the sensor of monitoring L. Subsequently, this research focuses on utilizing the signal strength vectors generated by monitoring sensors deployed in LoRaWAN networks as unique fingerprints for legitimate LoRaWAN devices. These signal fingerprint vectors, denoted as x_i, are collected in real-time and used to build a training set. Subsequently, a Support Vector Machine (SVM) classifier is trained, specifically the SCSVM classifier, to determine whether a given vector sample corresponds to a legitimate LoRaWAN device. This approach aims to enhance the security and authentication mechanisms within LoRaWAN networks by taking advantage of the distinctive characteristics of signal strength vectors.

4.5.1. Training set

We populate the training set with RSS vector samples from legitimate LoRaWAN devices. Each sample in the dataset includes the reception time from the transmitter to the monitoring sensor, the transmitter's media access control (MAC) address, and the corresponding RSS value. The training dataset Y is defined as shown in Eq. (14).

$$Y = \{y_1, y_2, y_2, \ldots, y_N\} \tag{14}$$

Where N represents the quantity of samples in the signal fingerprint vector y in the training dataset Y.

4.5.2. Single-class SVM classification

To measure RSS for each feature, first, it is identified the type of wireless signal to be measured, such as Wi-Fi. Utilize a signal receiver capable of detecting and reporting RSS values, which could be a specialized device like a spectrum analyzer, or software tools available on mobile devices or computers. Position the receiver at the desired location, and capture the RSS value using the receiver's appropriate software commands. For instance, on Linux systems, a command like 'iwconfig' is used to retrieve RSS values for each signal. Moreover, it is also ensured to record and store these values along with relevant metadata such as time and location for each measurement. The same process is repeated to gather comprehensive RSS data for each feature.

The primary goal of the SCSVM is to establish decision functions based on feature vectors derived from the training dataset. SCSVM operates by discerning potentially malicious devices through the identification of an optimal hyperplane within a nonlinear space. This objective is mathematically defined as a quadratic optimization problem, represented by Eq. (15). By formulating the problem in this manner, SCSVM aims to effectively distinguish between legitimate and malicious devices within the system.

$$\underset{\xi,\Xi,c}{min}\frac{1}{2}\|w\|^2 + \frac{1}{\mu N}\sum_{j=1}^{N}\Xi_j - c, \quad s.t.\phi(y_j) \geq c - \Xi_j, \Xi_j \geq 0 \tag{15}$$

Where, j ranges from 1 to N, N denotes the total number of instances, and $s.t.$ stands for "subject to". The function $\phi(\cdot)$ is employed as a nonlinear mapping function to process feature vectors within the training set. The weight vector for the model is represented as w, while ξ_i means nonzero slack variables introduced to accommodate tolerance within the model. A regularization parameter, μ, is set to 0.03 to control this tolerance within a specified range (0, 1). Within the training data, the vector sample y_j^*, subject to $\phi(y_j) - c = \Xi_j$, functions as the support vector located at the boundary of the decision function. c is the threshold or bias. These components collectively contribute to the formulation and operation of the model, which helps facilitate the delineation between various instances within the system.

$$G(y) = sgn\Big(w\rho(y) - c\Big) \tag{16}$$

$G(y)$ is the output of the decision function, y is the input feature vector, $sgn(\cdot)$ is the sign function, and $\rho(y)$ is the feature mapping function. In the scenario where $g(y) = 1$ means that the fingerprint vector for the strength of signal y originates from an authentic LoRaWAN device and $g(y) = -1$ indicates that it originates from a WiFi attacker.

To identify whether the test sample vector x_j lies within the hyperplane, the decision function uses a nonlinear kernel function $K(y_i, y_j)$, as described in Schuëller and Jensen [81] (Eq. (17)).

$$G_{SCSVM}\left(y_i\right) = sgn\left(\sum_{j=1}^{N} L\left(y_j, y_i\right) - c\right), \quad 0 < \eta_j < \frac{1}{\mu N} \tag{17}$$

Here, η_j represents the Lagrange multiplier derived through the utilization of the $\phi(\cdot)$ function to optimize the margin. μ is a regularization parameter that controls the trade-off between maximizing the margin and minimizing classification errors in support vector machine (SVM) models. Typically, three distinct kernel functions are employed in various situations, namely the polynomial function, the radial basis function (RBF) and the linear function, as described in Shang et al. [82] (Eq. (18)).

$$\begin{cases} K\left(y_j, y_i\right) = y_j^T y_i, \\ K\left(y_j, y_i\right) = \left(y_j^T y_i + 1\right)^s, \\ K\left(y_j, y_i\right) = e^{\left(-\alpha \| y_j - y_i \|^2\right)} \end{cases} \tag{18}$$

Here, s represents the degree parameter for the kernel of the polynomial function. It controls the degree of the polynomial in the polynomial kernel function.

The outcome of Eqs. (15)–(18) is the development of a robust SCSVM model that can effectively differentiate between legitimate and spoofed signals. By solving the quadratic optimization problem and applying the decision function in conjunction with appropriate kernel functions, the SCSVM accurately classifies legitimate and spoofed signals based on their signal characteristics, even in nonlinear and complex scenarios.

These equations collectively contribute to the SCSVM's capability to enhance network security by identifying and mitigating potential threats, thus ensuring the integrity of the LoRaWAN system.

- The linear kernel function is primarily employed when dealing with linearly separable data. Compared to polynomial and RBF kernel functions, it involves fewer parameters, which helps to reduce the computation speed. It yields excellent classification outcomes for linearly separable datasets.
- The polynomial function can transform a low-dimensional data space into a high-dimensional data space, although with a higher parameter count. Particularly with high polynomial orders, computational complexity escalates significantly, impeding practical computation.
- The RBF function proficiently maps samples to higher-dimensional spaces. It demonstrates superior performance in both large and small-sample training scenarios, with fewer parameters compared to the polynomial kernel function. Following a comparative analysis of the computational outcomes of such kernel functions, this research opts for the RBF function, attributed to its highest detection rate.

In this section, the utilization of the SVM algorithm is discussed to enhance the performance of the classifier based on the classification outcomes of the SCSVM when offline training data are accessible. SVM is described as a collection of kernel-based learning techniques for data classification, encompassing both the training and testing phases. Each sample instance within the training set is characterized by a class label and attributes (features). For example, in the context of countering spoofing attacks from WiFi to LoRaWAN, instances without spoofing attacks can be labeled as "+1", while instances with spoofing attacks can be labeled as "−1". This approach aims to leverage SVM's capabilities to refine the classification process and improve the accuracy of detecting spoofing attacks in wireless networks.

Training samples can be acquired through regular monitoring of network activities. A labeled training set $\mathbf{T}$ consisting of I feature vectors is defined by Eq. (19), where x_i represents the label of $\mathbf{y_i}$.

$$\mathbf{T} = \left\{ \mathbf{y_i} | \mathbf{y_i} \in R^N, x_i \in \{-1, +1\} \right\} \tag{19}$$

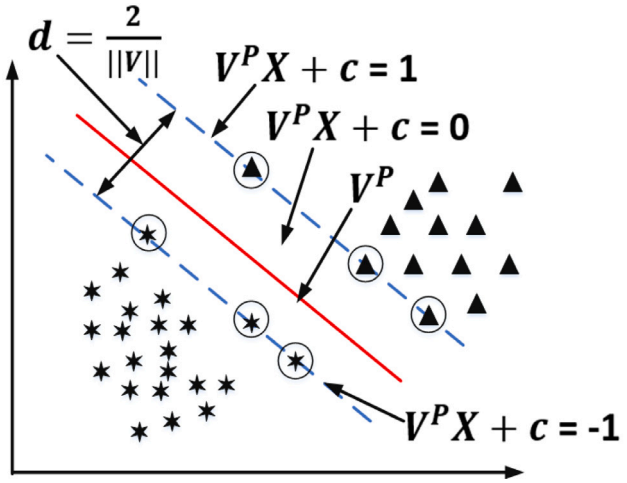


Fig. 7. Dataset with Linear separability.

Every feature vector y_i is a real vector of dimension N as given in Eq. (20).

$$y_i = \left[RSS_i^1, RSS_i^2, RSS_i^L, \ldots, \right]^P \tag{20}$$

Where P represents the total number of different RSS measurements taken for each feature vector y_i, while L denotes the number of RSS measurements per feature vector. Our objective is to optimize the margin hyperplane that separates the feature vectors with x_i equal to $+1$ and x_i equal to -1. The formulation of this hyperplane is given in Eq. (21).

$$\mathbf{V}^P X + c = 0 \tag{21}$$

$\mathbf{V}$ represents the normal vector of the hyperplane, c stands for the bias variable, and $\mathbf{X}$ denotes the feature vector of the sample located on the hyperplane, as illustrated in Fig. 7.

The objective is to select a hyperplane that maximizes the margin between positive and negative samples. This optimization is subject to the following constraint given in Eq. (22).

$$x_i\left(\mathbf{V}^P y_i + c\right) \geq 1, \forall_i \in [1, N] \tag{22}$$

As depicted in Fig. 8, when the training samples in the transformed space are not linearly separable, adjustments to the optimization problem can be made by incorporating non-negative slack variables $\Xi_i \geq 0$ as given in Eq. (23).

$$x_i\left(\mathbf{V}^P y_i + c\right) \geq 1 - \Xi, \qquad \forall_i \in [1, N] \tag{23}$$

The hyperplane $(\mathbf{V}^T)$ can be calculated by solving the optimization challenge in the fundamental form as given in Eq. (24).

$$\begin{aligned} \min_{\mathbf{V},c} \frac{1}{2}\|\mathbf{V}\|^2 + D\sum_{i=1} \Xi_i, s.t. x_i\left(\mathbf{V}^P y_i + c\right) \\ \geq 1 - \Xi_i, \forall \mathbf{y}_i \in T, \quad \Xi_i \geq 0 \end{aligned} \tag{24}$$

The dual form of Eq. (24) is given in Eq. (25).

$$\begin{aligned} & min \frac{1}{2} \sum_{i=1,j=1} \beta_i \cdot \beta_j x_i \cdot x_j \cdot y_i^P \cdot y_j \\ & - \sum_{i=1}^{\infty} \beta_i, s.t. \sum_{i=1} \beta_i \cdot x_i = 0, \quad 0 \leq \beta_i \leq D \end{aligned} \tag{25}$$

Given that $D > 0$ represents a compromise parameter balancing error and margin, nonlinear SVM problems can be addressed using "kernel tricks". If the kernel function is denoted by $K(y_i, y_j) = \phi(y_i)^P \phi(y_j)$,

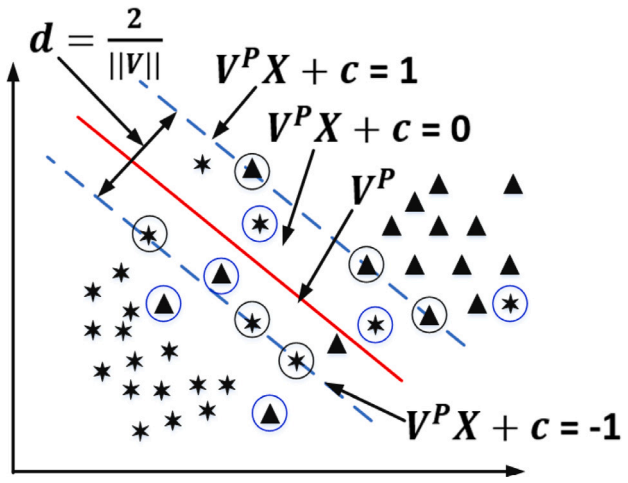


Fig. 8. Dataset without linear separability.

then Eq. (25) becomes:

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{N} \sum_{i=1}^{N} \beta_i \cdot \beta_j x_i \cdot x_j K(y_i, y_j) - \sum_{i=1}^{N} \beta_i, \quad s.t. D \geq \beta_i \geq 0, \quad \sum_{i=1}^{N} \beta_i x_i = 0 \tag{26}$$

For testing purpose, the following kernel was utilized:

$$\begin{cases} K(y_i, y_j) = y_i^P x_j, \\ K(y_i, y_j) = \left(x_i^P x_j + 1\right)^s \\ K(y_i, y_j) = tanh\left(\beta x_i^P x_j + d\right) \\ K(y_i, y_j) = e^{\left(-\alpha \|y_i - y_j\|^2\right)} \end{cases} \tag{27}$$

Furthermore, when provided with a signal strength vector sample y, the decision function is articulated as follows:

$$G_{SVM}(y) = sgn\left(\sum_{i=1}^{N} \beta_i x_i K(y_i, y) + c\right) \tag{28}$$

Here, $G_{SVM}(y)$ is employed to identify the test sample **t**. If $f_{SVM}(y)$ yields a value of 1, it suggests that the test sample **t** is associated with a genuine LoraWan device, implying the absence of any spoofing attack. Conversely, if $f_{SVM}(y)$ equals −1, it signifies that the test sample **t** originates from a WiFi attacker, indicating the presence of a spoofing attack on the network.

A multi-class SVM is constructed by combining multiple binary SVM classifiers. In the proposed work, $\binom{N}{2}$ binary classifiers are trained. Each distinguishing between a pair of classes, and the final prediction is determined by the class that receives the highest cumulative score. This method allows SVM to effectively handle multi-class classification by aggregating results from several binary classifications.

In everyday wireless communication, spoofing attacks are relatively infrequent compared to genuine communications. Therefore, while it is easy to gather a substantial volume of legitimate communication samples, obtaining spoofing attack samples is challenging due to their scarcity and the diverse methods employed for implementation [83]. Traditional classification algorithms typically necessitate an equal number of samples from both classes. However, due to the limited availability of spoofing attack training data, this poses a significant challenge. In this research, to address this issue, the SCSVM algorithm is utilized to train on legitimate data and construct a classifier capable of detecting spoofing attacks. As more spoofing attack samples are identified by the SCSVM classifier, further enhancements are explored using the SVM algorithm to refine the classifier's performance.

In the proposed research, SVM is preferred over neural networks for decision-making due to their simplicity and ability to generalize well with limited data. For a fair comparison, all machine learning models including Naive Bayes (NB), Random Forest (RF), Logistic Regression (LR), SVM, and the Convolutional Neural Network (CNN) baseline are trained and evaluated on the same dataset under identical experimental conditions. The neural network consisted of three fully connected layers using ReLU activation and Adam optimization, trained on the same training and testing splits as the SVM. The SVM model, configured with an RBF kernel and regularization parameter (μ = 0.03), consistently outperformed the neural network in terms of accuracy and stability, particularly when the training data volume was limited.

Further, SVM can achieve accuracy improvements of up to 10% over neural networks when trained on small datasets (e.g., fewer than 1000 samples) [84,85]. Moreover, SVM provides a clear geometric interpretation in order to find the optimal hyperplane that separates classes in feature space. Additionally, SVM indirectly performs feature space transformation through kernel functions such as polynomial, linear, and sigmoid. This allows them to handle non-linear decision boundaries effectively. SVMs with non-linear kernels can improve classification accuracy by up to 20% in complex datasets compared to linear models [86]. SVM is also robust to noise in the training data and requires fewer computational resources for training compared to neural networks. It often needs 50% less training time on comparable hardware [87]. While neural networks outperform at handling complex tasks when computational resources are available, SVM still remains a valuable choice for tasks with smaller datasets.

The detection accuracy of the proposed model is largely governed by three mathematical components.

1. Feature derivation equations for shimmer and jitter (See Table 3), which quantify timing and amplitude perturbations in received signals;
2. RSS-based spatial correlation models (Eqs. (1)–(3)), which capture environmental variations and strengthen feature discrimination.
3. Optimization functions in the S-CSVM (Eqs. (15)–(18)), where the margin and regularization parameter (μ = 0.03) directly affect classification sensitivity.

These components collectively determine the model's ability to distinguish authentic from spoofed signals with high precision.

5. Experimental setup, design, result and analysis

In this section, the performance of the proposed model in detecting spoofing is tested within heterogeneous networks using HackRF One and LoRaWAN devices (specifically, Pycom LoPy4 nodes). Pycom LoPy4 is a highly versatile LoRaWAN node that supports multiple wireless protocols, including LoRa, Wi-Fi, and Bluetooth. To evaluate the performance of the proposed model, all the analysis are carried on 8 GB RAM, NVIDIA GeForce RTX Series such as RTX 3060.

The scenario depicted involves two Pycom LoPy4 nodes engaged in regular data communication. Additionally, the HackRF One device, which emulates a WiFi signal transmitter, has the capability to transmit data directly to the LoRaWAN device. This setup allows the HackRF One to mimic a legitimate LoRaWAN device and execute spoofing attacks on other LoRaWAN devices.

The spoofing detection experiments proposed are conducted in an indoor setting. Illustrated in Fig. 8, a WiFi device positioned approximately 3 m from a legitimate LoRaWAN device initiates a spoofing attack. While the LoRaWAN transmitter and receiver are engaged in communication, a malicious WiFi device in close proximity can intercept LoRaWAN packets, posing as a LoRaWAN transmitter to execute a

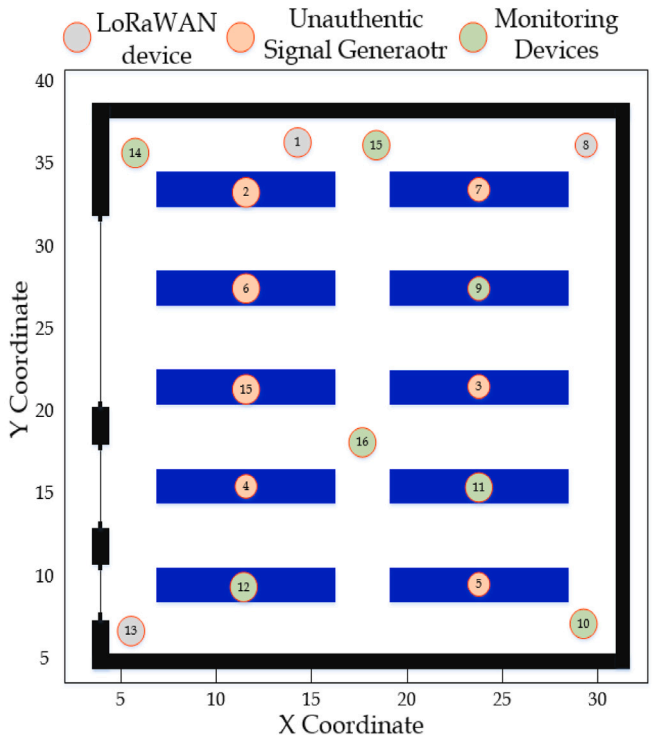


Fig. 9. Network topology for the experimentation of the proposed work.

CTC spoofing attack on the LoRaWAN receiver. This scenario represents the targeted spoofing attack. In Fig. 9, 16 testing locations, denoted by dots, cover an area of 30×40 m^2. Two spoofing attack scenarios were assessed. The first scenario involves WiFi attackers within the room, with ten locations designated for legitimate LoRaWAN devices and the remaining ten for attackers. The HackRF One, situated at locations 9–16, moves between the ten locations, launching spoof attacks on LoRaWAN devices. Detection involved deploying four sensors to measure RSS of audible frames, collecting 200 packet-level RSS samples at each location. Standard RSS samples from locations 1–10 were utilized to train the SCSVM classifier for detecting spoofing attacks from locations 9–16. In the second scenario, WiFi devices with extensive deployment and wider transmission range outside the room were considered. Data from all test locations (i.e., locations 1–8) are used to train the SCSVM classifier.

5.1. Analysis of signal strength

Fig. 10 illustrates the data distribution across 20 locations captured by 12 monitoring sensors. A total of 6000 signal samples are received (300 collected from each location). In particular, such signals exhibited oscillation for stationary devices, with closely clustered RSS values at the same position. Various factors, such as multipath effects and obstacles, contribute to this oscillation, particularly over longer distances between the sender and receiver. To mitigate this influence, additional samples are collected at each location, followed by employing data cleaning techniques.

The corresponding probability histogram of the signals received in 20 locations is depicted in Fig. 11. Existing research suggests that signal samples from a given sensor typically conform to a Gaussian distribution, although non-Gaussian distributions can often a mixture of multiple Gaussian distributions. Fig. 11 further supports this observation, revealing a mixture of Gaussian distributions across the sampled locations. For instance, in Fig. 11(a), four Gaussian distributions are discernible, while in Fig. 11(b), two Gaussian distributions are evident. Each subplot illustrates the signal strength range, typically between −110 db and −10 db, with calculated mean ν and standard deviation σ values for each location.

A signal with an RSS of −100 dB can be detected, provided that the receiver has sufficient sensitivity, typically specified in its technical details. To detect such a weak signal, use a receiver with a sensitivity rating that can detect signals at or below −100 dB. It is ensured that the measurement environment is free from significant interference and noise, as high noise levels can obscure weak signals. Then employed a high-gain antenna to enhance signal reception, and position it optimally. Additionally, signal processing techniques such as the low noise amplifier and noise reduction [88,89] are used to improve the signal-to-noise ratio and make the weak signal more discernible.

5.2. Comparison of different machine learning algorithms

In order to assess the efficacy of the model proposed in practical scenarios, various machine learning algorithms including Naive Bayes, Random Forest (RF), Linear Regression (LR), SVM, and SCSVM are selected for comparative experiments. The evaluation of the performance of the proposed model involve simulating ten attack locations (designated as locations 9–16 in Fig. 9).

Additionally, a spoofing monitoring program is implemented, wherein the distance between the legitimate LoRaWAN device and the WiFi attacker are categorized into four ranges: less than 3 m, 3 to 4 m, and 4 to 6 m, and 6 to 8 m, respectively. In particular, it is observed that the performance of the proposed method improved with increasing attack distance. Specifically, when the attack distance exceeded 7 m, the spoofing detection method demonstrated remarkably high detection performance.

This section introduces the comparison between the performance of different machine learning algorithms for the proposed model in detecting and mitigating spoofing attacks in real-world scenarios as given in Table 4.

When the distance between the legitimate LoRaWAN device and the WiFi attacker is less than 3 m, the SVM method exhibits an average accuracy rate of 92.86%, with a standard deviation of 11.03. Similarly, the LR method demonstrates an average accuracy rate of 90.02% with a standard deviation of 10.33%. The values provided in Table 4 reveals that as distance increases, the accuracy of all methods improves. In particular, the logistic regression algorithm exhibits suboptimal performance, particularly when the distance is less than 2 m, with an average accuracy rate of only 68.13%. Conversely, the OSVM algorithm consistently outperforms the other methods at all distances, achieving accuracy rates that exceed 95%.

In the context of small-distance spoofing attacks (less than 1.5 m), the SCSVM algorithm shows significantly higher accuracy compared to the other four algorithms. Moreover, its performance superiority persists as the distance increases to 2–3 meters and 3–5 meters. Notably, the standard deviation of the accuracy of the SCSVM algorithm remains consistently smaller than that of the other methods, indicating greater stability in its detection performance. Therefore, the SCSVM algorithm emerges as the most effective and stable detection method across different test distance scenarios, outperforming NB, RF, LR, SVM, and CNN algorithms.

It can be seen from Table 4 that higher accuracy in detecting spoofing attacks from further distances, such as 99.23% for distances between 6 m and 8 m, compared to 96.80% for distances less than 3 m, is due to several factors. At greater distances, the spoofed signal undergoes more propagation effects, such as attenuation, additional noise, and multipath fading, which makes its features more pronounced and easier to distinguish from the original spoofed signal. Moreover, there are more differences in power levels and timing between the spoofed and real signals when they are farther apart. This makes it easier to spot the nature of the received signals. Apart from that, environmental factors, such as obstacles and interference, also affect

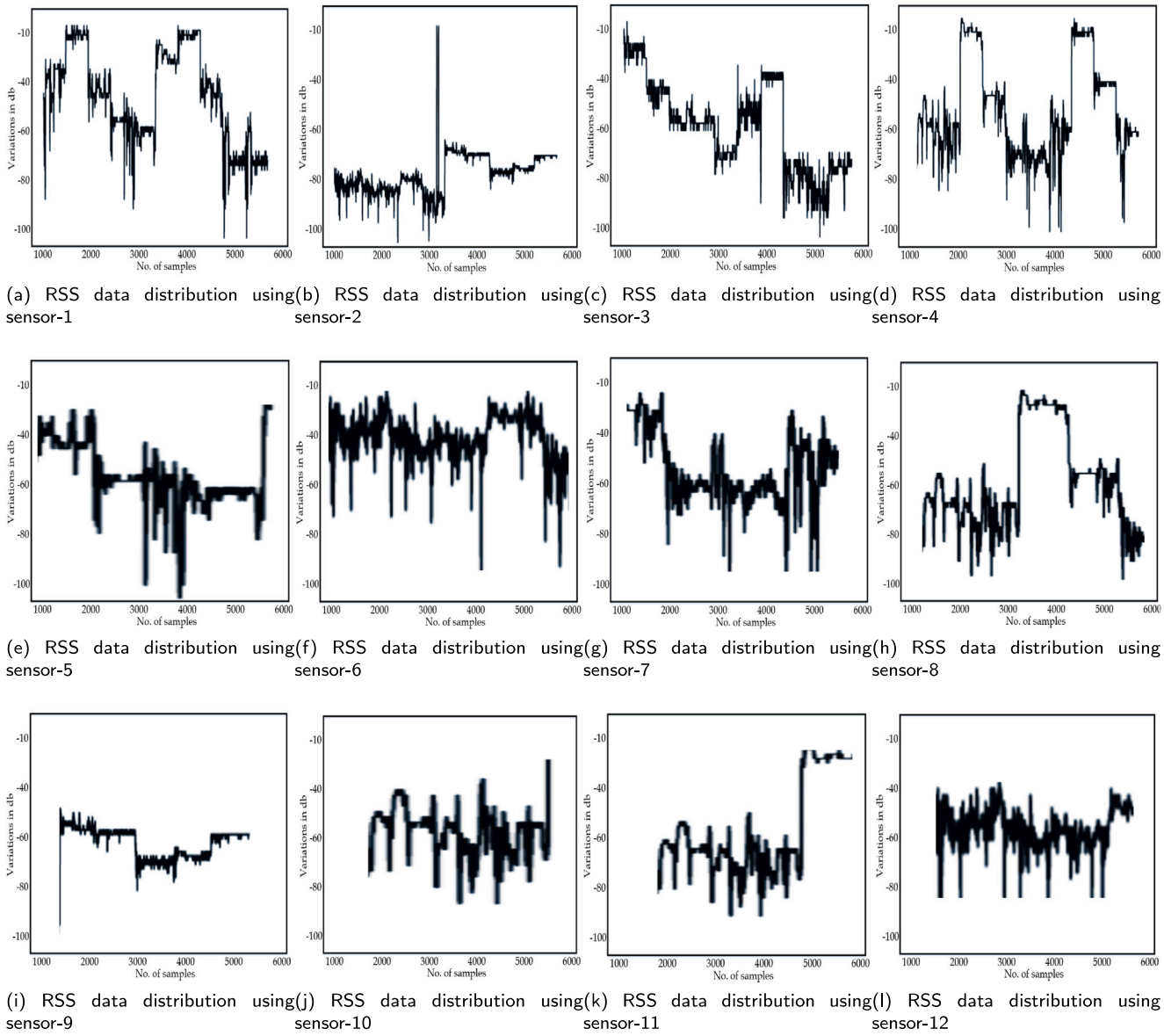


Fig. 10. The RSS data dispersion at 20 locations.

spoofed signals more significantly at longer ranges, which ultimately enhances the ability of the system to identify spoofing attempts.

Moreover, the lower standard deviation indicates the system is highly accurate and more robust. As shown in Table 4, SCSVM exhibits lower standard deviations and high accuracy values.

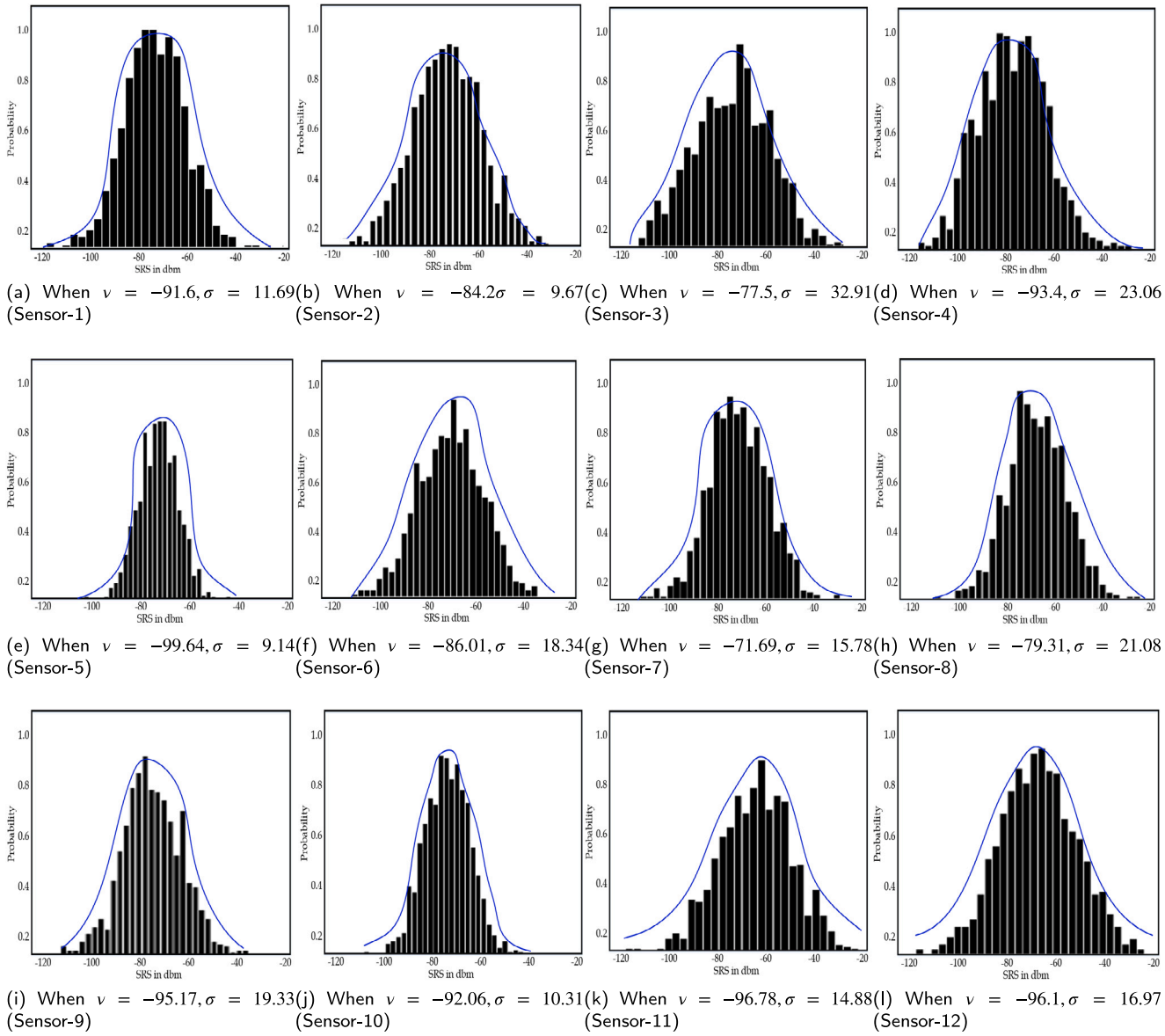
Further, it can found in Table 4 that the high standard deviation of up to 12% to 14% in the accuracy of the trained classifiers indicates significant variability in their performance. Specifically, due to data variability and training sample differences. The data used in the proposed work has various signal samples, and each sample in the dataset varies from each other because of each sample has different feature values. Additionally, differences in training subsets, such as those encountered in cross-validation, result in accuracy variations. Moreover, the sensitivity of SVM to hyperparameters and the polynomial kernel also cause accuracy to vary widely.

To mitigate the risk of overfitting, especially given the single-class training approach, multiple safeguards are adopted. First, the dataset is split into distinct training and testing subsets collected across different days, environments, and distances, ensuring the model is evaluated on unseen data. Second, the S-CSVM regularization parameter (μ = 0.03) is applied to control margin tolerance and avoid bias toward training samples. Third, evaluation metrics (precision, recall, F-score, AUC) also supported the robustness of the classifier. The stability of accuracy across different testing conditions (Table 4) also shows that the reported high performance is not due to overfitting but rather to the discriminative strength of the selected features.

5.3. Ablation study

The ablation results presents in this section demonstrate that (a) timing and amplitude variations captured through Jitter and Shimmer act synergistically, (b) RSS features contribute environment-specific resilience, and (c) frequency-modulation parameters add a smaller but consistent improvement. The performance gains persist across all evaluated distances, with the largest improvements observed for 6–8 m scenarios, confirming the robustness of the proposed feature design.

To evaluate the contribution of individual feature families, an ablation study is conducted under identical experimental settings, using the same training and testing splits, z-score normalization, and S-CSVM with an RBF kernel and regularization parameter μ = 0.03. Eight configurations are evaluated: (i) RSS only, (ii) Jitter only, (iii) Shimmer only, (iv) Jitter + Shimmer, (v) RSS + Jitter, (vi) RSS + Shimmer,



(a) When $\nu = -91.6, \sigma = 11.69$ (Sensor-1) (b) When $\nu = -84.2 \sigma = 9.67$ (Sensor-2) (c) When $\nu = -77.5, \sigma = 32.91$ (Sensor-3) (d) When $\nu = -93.4, \sigma = 23.06$ (Sensor-4)

(e) When $\nu = -99.64, \sigma = 9.14$ (Sensor-5) (f) When $\nu = -86.01, \sigma = 18.34$ (Sensor-6) (g) When $\nu = -71.69, \sigma = 15.78$ (Sensor-7) (h) When $\nu = -79.31, \sigma = 21.08$ (Sensor-8)

(i) When $\nu = -95.17, \sigma = 19.33$ (Sensor-9) (j) When $\nu = -92.06, \sigma = 10.31$ (Sensor-10) (k) When $\nu = -96.78, \sigma = 14.88$ (Sensor-11) (l) When $\nu = -96.1, \sigma = 16.97$ (Sensor-12)

Fig. 11. The probability histogram of RSS at 20 locations.

(vii) RSS + Jitter + Shimmer, and (viii) Full (RSS + Jitter + Shimmer + FM). The results for the full configuration are presented in Table 7. Each model is trained and tested on identical data partitions, and performance is assessed using the evaluation metrics such as accuracy, and the metrics mentioned in Section 5.5.

Table 5 provides the quantitative results. A performance progression is observed as additional feature groups are introduced. Models based on single-feature families (i–iii) achieve moderate performance which shows that individual cues such as timing or amplitude irregularities alone are insufficient for reliable spoofing detection. Combining Jitter and Shimmer (iv) improves results substantially, showing that timing and amplitude perturbations provide more information. The inclusion of RSS (v–vii) further enhances robustness by introducing environmental context that helps differentiate genuine and spoofed signals. The configuration using all three primary feature groups (G) achieves near-optimal performance, while the full model (H), which also includes frequency-modulation features, yields the highest overall accuracy, precision, and stability across all metrics as given in Table 7.

All features are extracted from the same time windows and synchronized timestamps. When combining feature groups, each feature is standardized using statistics computed from the training data and applied to the corresponding test set. Moreover, the hyperparameters are fixed across all experiments to ensure comparability. To prevent bias due to varying device distances, additional stratified evaluations are performed by distance bins, following the same segmentation used in Table 4.

5.4. Computational complexity analysis

The computational time results highlights that there are significant variances in average test times, σ, and the range of test times (min to max). The K-Means, K-Nearest Neighbors (KNN), Logistic Regression (LR), Random Forest (RF), and SCSVM, with respective average test times noted as 0.12666 s, 0.22578 s, 0.31306 s, 2.01943 s, and 0.021941 s, respectively.

Table 6 reveals SCSVM as the fastest method, with an average test time of approximately 0.022 s, substantially outpacing the second quickest, K-Means, by about 100 ms. The slowest among the evaluated methods is Random Forest, with an average test time exceeding 2 s. This substantial difference in processing times shows the superior computational speed of the SCSVM method in detecting spoofing attacks

Table 4
Comparison of different machine learning algorithms in terms of accuracy and std.

Testing distance	Algorithms	Accuracy	std	Min	Max
When distance: < 3 m	Naive Bayes	83.26	10.38	85.16	81.36
	RF	87.94	11.34	89.39	86.49
	LR	90.02	10.33	89.93	90.11
	SVM	92.86	11.03	90.36	95.36
	CNN	94.65	8.09	93.67	96.77
	SCSVM	96.80	4.68	96.45	97.15
When distance: > 3 m and < 4 m	Naive Bayes	91.07	11.99	89.99	92.16
	RF	83.34	12.03	79.99	86.69
	LR	90.79	9.49	89.90	91.68
	SVM	91.81	10.33	91.64	91.99
	CNN	93.13	8.90	92.41	93.09
	SCSVM	98.38	2.16	97.89	98.88
When distance: > 4 m and < 6 m	Naive Bayes	96.92	10.36	96.66	97.19
	RF	95.74	11.36	95.16	96.33
	LR	94.85	10.37	94.69	95.01
	SVM	93.68	9.49	91.67	95.69
	CNN	94.81	8.55	93.22	96.92
	SCSVM	98.46	2.08	97.99	98.94
When distance: > 6 m and < 8m	Naive Bayes	92.69	12.36	91.69	93.69
	RF	93.18	11.69	91.67	94.69
	LR	95.74	8.96	95.16	96.33
	SVM	94.92	10.36	94.16	95.69
	CNN	96.22	5.09	96.45	97.88
	SCSVM	99.22	1.67	98.79	99.68

Table 5
Ablation over feature subsets (identical splits, S-CSVM, $\mu = 0.03$).

Features	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	AUC (%)
(i) RSS only	94.12	95.03	94.67	94.84	96.20
(ii) Jitter only	92.87	93.56	92.34	92.94	95.02
(iii) Shimmer only	93.14	94.05	93.72	93.88	95.31
(iv) Jitter + Shimmer	95.48	96.32	95.79	96.05	97.22
(v) RSS + Jitter	96.73	97.14	96.69	96.91	98.12
(vi) RSS + Shimmer	97.02	97.53	97.11	97.32	98.33
(vii) RSS + Jitter + Shimmer	98.66	99.05	98.71	98.88	99.10

Table 6
Computational time analysis.

	Time (s)	Naive Bayes	RF	LR	SVM	CNN	SCSVM
Proposed Model	Max time	0.069	0.075	0.762	0.069	0.0045	0.0025
	Min time	0.051	0.066	0.531	0.055	0.0031	0.0011
	Average time	0.06	0.070	0.646	0.062	0.0038	0.0018
Meng et . al [91]	Max time			0.03			
	Min time			0.02			
	Average time			0.025			
Panice et . al [90]	Max time			0.0021			
	Min time			0.0010			
	Average time			0.0015			
Shafique et . al [92]	Max time			0.010			
	Min time			0.023			
	Average time			0.014			

that suggests its potential for immediate detection with minimal latency compared to the other machine learning algorithms evaluated.

For comparison purposes, several existing models are implemented on the same platform as our proposed model. The computational complexity of both the existing models and the proposed model is compared. It is found from the results that the model presented in Panice et al. [90] slightly outperforms our proposed model in terms of computational time.

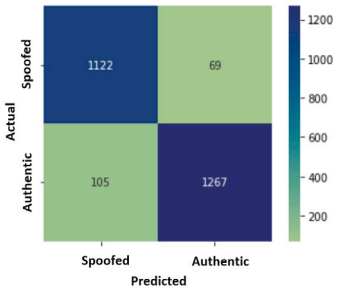
5.5. Other statistical evaluation parameters

In this section, the other common evaluation parameters are discussed in order to gauge the performance of the proposed model for signal spoofing detection. The evaluation parameters include F-score, precision, recall, detection rate, and area under the curve (AUC). The mentioned parameters other than AUC can be calculated using confusion matrix. The confusion matrices for the proposed model are displayed in Fig. 12.

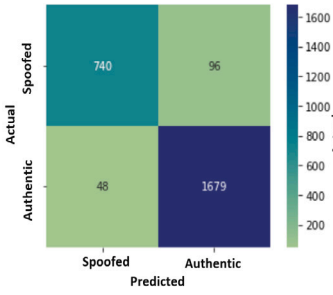
5.5.1. Precision

Precision is used in binary classification tasks that measures the accuracy of positive predictions made by a model. Mathematically, it can be calculated using Eq. (29).

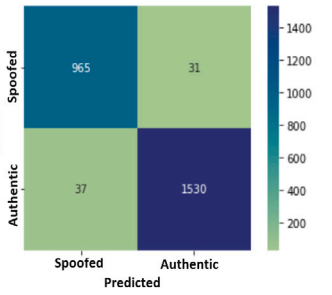
$$Precision = \frac{T.P}{T.P + F.P} \tag{29}$$



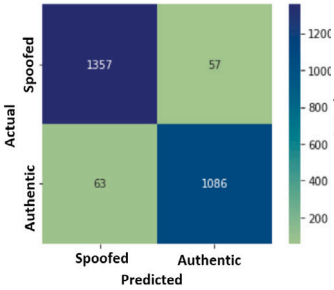
(a) When NB is incorporated



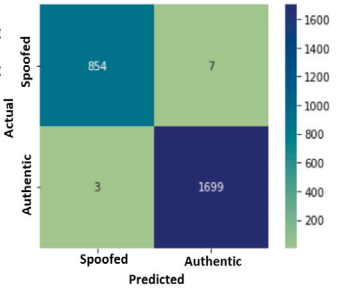
(b) When RF is incorporated



(c) When SVM is incorporated

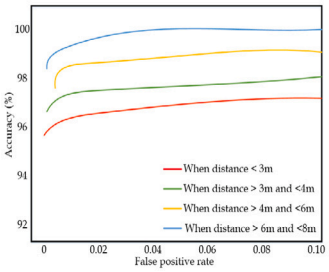


(d) When LR is incorporated

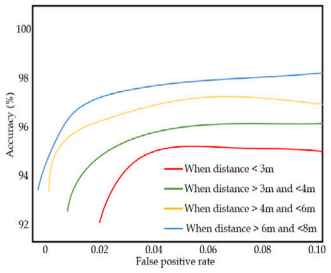


(e) When SCSVM is incorporated

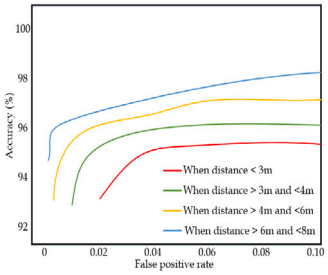
Fig. 12. Confusion matrices for the proposed model when different machine learning algorithms are incorporated.



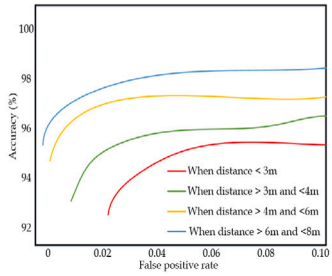
(a) AUC for SCSVM



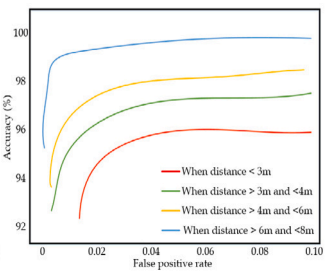
(b) AUC for Naive Bayes



(c) AUC for RF



(d) AUC for LR



(e) AUC for SVM

Fig. 13. The AUC curves at different locations for various machine learning algorithms.

5.5.2. Recall

Recall measures the ability of a model to correctly identify all relevant instances of the positive class from the total number of actual positive instances in the dataset. It can be calculated using Eq. (30).

$$Recall = \frac{T.P}{T.P + F.N} \tag{30}$$

5.5.3. F-score

The F-score is used to evaluate the performance of a binary classification model. It combines the precision and recall of the model into a single value. Mathematically, it can be calculated using Eq. (31).

$$F_\gamma = \frac{(1+\gamma)^2 \times precision \times recall}{\gamma^2 \times precision + recall} \tag{31}$$

Here, γ is a parameter that adjusts the emphasis between precision and recall. $\gamma = 1$ is commonly used to calculate F1 score.

Table 7 provides the comparison of the proposed machine learning model with the existing models. To ensure a fair comparison, the existing model is implemented on the same platform as the proposed model. From Table 7, it can be seen that the proposed model when SCSVM is incorporated, the accuracy is better by approximately 3.45% of the existing models. Which also shows the effectiveness of the proposed model over the existing ones.

Furthermore, it can be seen from Table 7 that the accuracy of the proposed model is slightly greater than 99%. An accuracy of > 99% does raise concerns about overfitting if the model is tailored to the training data and may not generalize well to new data. To address this, the system is evaluated using the data, which is collected on different days in various conditions, such as in bright sunlight and in darkness, with a sufficiently significant separation from the training data. By testing the model on unseen data, especially from different days under different conditions, it is ensured that the high accuracy is not simply a result of memorizing the training examples. This testing process helps to confirm that the model can effectively generalize its learned patterns to new and unseen situations.

5.5.4. Area under the curve

AUC is a graphical representation of the trade-off between the true positive rate (sensitivity) and the false positive rate (1 - specificity). Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular threshold.

The AUC quantifies the overall performance of a binary classification model in all possible threshold settings. It ranges from 0 to 1, where a value of 1 indicates perfect classification, and a value of 0.5 suggests that the model does not perform better than random guessing. A higher AUC score indicates better discrimination between positive and negative classes, with larger areas under the ROC curve representing stronger model performance as shown in Fig. 13.

Table 7
Comparison of the statistical values for the proposed model when incorporating Naive Bayes (NB), RF, LR, SVM, and SCSVM.

ML Algorithms →	NB	RF	SVM	LR	SCSVM
Proposed					
Accuracy	93.20	94.37	97.34	95.31	99.20
Precision	95.15	95.78	96.66	97.83	99.99
Recall	97.16	98.64	98.01	98.64	99.87
F-score	96.66	98.18	97.84	99.67	99.94
Existing model: De et al. [93]					
Accuracy	93.67	95.15	96.15	95.55	97.54
Precision	95.64	96.15	95.15	96.66	97.82
Recall	95.67	96.89	98.68	98.16	97.84
F-score	98.64	97.65	97.89	98.46	98.44
Existing model: Sun et al. [94]					
Accuracy	97.45	96.67	96.45	97.45	98.88
Precision	93.45	92.87	97.66	97.45	98.15
Recall	94.54	95.64	95.15	96.33	98.46
F-score	95.65	98.61	98.66	97.86	97.99
Existing model: Ying et al. [95]					
Accuracy	94.66	95.15	96.48	96.11	97.33
Precision	92.69	94.68	97.62	96.54	97.89
Recall	95.64	93.65	94.65	95.68	97.68
F-score	95.64	97.66	95.64	95.33	96.67

6. Real-world application and deployment perspective

Although the experiments are performed under controlled laboratory conditions, the setup is carefully designed to emulate signal behavior observed in UAV operations. Authentic and spoofed signals are recorded across ten independent sessions using variable distances between transmitter and receiver (3–8 m), antenna tilts ranging from 0° to 45°, and both daylight and low-light conditions. These variations simulate the fluctuations a UAV experiences during attitude changes and minor GNSS multipath effects. Under these emulated flight conditions, the proposed S-CSVM framework consistently achieved an average detection accuracy of 99.22%, precision of 99.99%, recall of 99.87%, F1-score of 99.94%, and AUC of 99.4%. Even when the signal-to-noise ratio (SNR) is degraded by −3 dB to −5 dB, performance remained above 97% which demonstrates that strong robustness to moderate interference and path-loss variation.

The architecture of the system is well-suited for direct integration into UAV flight controllers. The shimmer, jitter, RSS, and FM features are extracted from received GPS/LoRaWAN telemetry frames without requiring access to raw I/Q data, which significantly reduces computational load. Implementation tests on a Raspberry Pi 4 (1.5 GHz quad-core, 4 GB RAM) showed that each 200 ms signal window can be processed and classified within 4.6 ms on average. This allows near-real-time inference at 200 Hz with negligible latency (< 10% of the typical 50 Hz control-loop period). This shows that the model can run as an onboard software module without modifying UAV hardware or navigation circuitry.

For practical field deployment, the feature-extraction module can be embedded as part of the UAV's telemetry pipeline, where each processed window yields a spoofing-likelihood score. When the S-CSVM detector identifies a probable spoofing event (confidence > 0.9), the UAV can trigger mitigation routines such as waypoint hold, altitude freeze, or fallback to inertial navigation. Integrating these actions within the autopilot firmware would enable autonomous resilience against spoofing without reliance on external communication links.

In addition to detection accuracy, onboard resource constraints are also considered to assess deployment feasibility on UAV hardware. The proposed framework is computationally lightweight, with the complete feature extraction and S-CSVM classification requiring approximately 4–5 ms per 200 ms signal window when executed on an ARM-based processor (1.5 GHz, 4 GB RAM). This corresponds to a processing throughput of about 200 Hz, sufficient for real-time detection within typical UAV control loop frequencies (20–50 Hz). The system operates efficiently and adds less than 2% to total power consumption during sustained operation. These results indicate that the model can be embedded in UAVs onboard computers without adversely affecting flight duration or control responsiveness.

Furthermore, for data authentication, the proposed model is trained only on authentic signal samples following the SCSVM paradigm, which is specifically designed for anomaly detection when spoofing samples are unavailable. To evaluate robustness, spoofed signals are not synthetically simulated but emulated using HackRF One devices to ensure that the test data accurately reflect real-world spoofing features under controlled laboratory conditions. Although this approach strengthens realism and reliability, training only on authentic data can constrain the generalization of the model to completely unseen or highly sophisticated spoofing types.

Future work will include field validation using multi-rotor UAVs equipped with dual-antenna GNSS receivers and onboard HackRF-based spoofing simulators. Planned trials will examine detection reliability under variable altitudes (10–100 m), dynamic yaw/pitch rotations, and outdoor multipath reflections from building facades. Comparative testing across altitudes will also quantify sensitivity to Doppler shifts and time-delay distortions introduced by genuine motion. The outcomes will be benchmarked against existing detection algorithms, providing empirical evidence of real-world transferability.

While the current implementation demonstrates high accuracy in controlled conditions, the system's performance under high-speed flight and in dense urban environments may exhibit slight degradation due to rapid SNR transitions and interference from adjacent RF sources. These factors will be addressed through adaptive windowing and online retraining strategies in future work.

7. Conclusion

This research introduces a novel approach to improving the security of autonomous systems against GPS spoofing attacks using machine learning techniques, particularly the S-CSVM classifier. The method effectively detects and mitigates spoofing attempts by analyzing both GNSS and LoRaWAN signals, incorporating features such as frequency modulations, shimmer, and jitter to distinguish between authentic and spoofed signals. A private dataset was collected and segmented based on the statistical characteristics of these features. The model intelligently identifies whether a received signal is legitimate or spoofed, allowing the UAV to proceed only when verification is successful. Model performance is validated through metrics such as accuracy,

precision, recall, F-score, and AUC, under various scenarios involving different distances between LoRaWAN devices and potential threats. Results show improved detection accuracy as the distance between LoRaWAN devices and the monitoring device increases, with the S-CSVM classifier achieving 99.22% accuracy at 6–8 meters. The model also operates in under one second, supporting real-time applications. Despite its effectiveness, limitations include reliance on high-quality GNSS data, sensitivity to signal fluctuations, and integration challenges in real-world environments. The proposed model is trained only on authentic data following the SCSVM paradigm, which is well-suited for anomaly detection scenarios where spoofed samples are scarce. During evaluation, spoofing signals are emulated using HackRF One devices to replicate real-world spoofing behavior under controlled conditions rather than relying on purely synthetic data.

CRediT authorship contribution statement

Arslan Shafique: Writing – original draft, Software, Methodology, Conceptualization. **Abid Mehmood:** Formal analysis, Data curation. **Moatsum Alawida:** Resources, Project administration, editing. **Shehzad Ashraf Chaudhry:** Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Abu Dhabi University's Office of Research and Sponsored Programs under Grant 19300961.

Data availability

Data will be made available on request.

References

[1] H. Ghaderi, Wider implications of autonomous vessels for the maritime industry: Mapping the unprecedented challenges, in: Advances in Transport Policy and Planning, vol. 5, Elsevier, 2020, pp. 263–289.
[2] G. Lykou, D. Moustakas, D. Gritzalis, Defending airports from UAS: A survey on cyber-attacks and counter-drone sensing technologies, Sensors 20 (12) (2020) 3537.
[3] A. Mehmood, A. Shafique, M. Alawida, A.N. Khan, Advances and vulnerabilities in modern cryptographic techniques: A comprehensive survey on cybersecurity in the domain of machine/deep learning and quantum techniques, IEEE Access 12 (2024) 27530–27555.
[4] G.K. Pandey, D.S. Gurjar, H.H. Nguyen, S. Yadav, Security threats and mitigation techniques in UAV communications: A comprehensive survey, IEEE Access 10 (2022) 112858–112897.
[5] A.E. Omolara, M. Alawida, O.I. Abiodun, Drone cybersecurity issues, solutions, trend insights and future perspectives: a survey, Neural Comput. Appl. 35 (31) (2023) 23063–23101.
[6] A. Shafique, A. Mehmood, M. Elhadef, Survey of security protocols and vulnerabilities in unmanned aerial vehicles, IEEE Access 9 (2021) 46927–46948.
[7] J.-P. Yaacoub, H. Noura, O. Salman, A. Chehab, Security analysis of drones systems: Attacks, limitations, and recommendations, Internet Things 11 (2020) 100218.
[8] A. Shafique, M.U. Rehman, K.H. Khan, S.S. Jamal, A. Mehmood, S.A. Chaudhry, Securing high-resolution images from unmanned aerial vehicles with dna encoding and bit-plane extraction method, IEEE Access (2023).
[9] E.B. Pohl, M.O. Verzier, M. Shoshan, Drone: Unmanned. Architecture and Security Series, vol. 1, dpr-barcelona, 2018.
[10] H. Benkraouda, E. Barka, K. Shuaib, et al., Cyber-attacks on the data communication of drones monitoring critical infrastructure, Comput. Sci. Inf. Technol. 8 (17) (2018) 83–93.
[11] G. Vasconcelos, R.S. Miani, V.C. Guizilini, J.R. Souza, Evaluation of dos attacks on commercial wi-fi-based uavs, Int. J. Commun. Netw. Inf. Secur. 11 (1) (2019) 212–223.
[12] K. Hartmann, C. Steup, The vulnerability of UAVs to cyber attacks-an approach to the risk assessment, in: 2013 5th International Conference on Cyber Conflict, CYCON 2013, IEEE, 2013, pp. 1–23.
[13] J. Xu, S. Ying, H. Li, Gps interference signal recognition based on machine learning, Mob. Netw. Appl. 25 (6) (2020) 2336–2350.
[14] D. Sathyamoorthy, M.M. Faudzi, M.Z. Fitry, Evaluation of the effect of radio frequency interference on global positioning system (GPS) accuracy via GPS simulation, Def. Sci. J. 62 (5) (2012) 338.
[15] H. Pirayesh, H. Zeng, Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey, IEEE Commun. Surv. Tutorials 24 (2) (2022) 767–809.
[16] A. Mpitziopoulos, D. Gavalas, An effective defensive node against jamming attacks in sensor networks, Secur. Commun. Netw. 2 (2) (2009) 145–163.
[17] M. Strasser, B. Danev, S. Čapkun, Detection of reactive jamming in sensor networks, ACM Trans. Sens. Netw. (TOSN) 7 (2) (2010) 1–29.
[18] S.P. Arteaga, L.A.M. Hernández, G.S. Pérez, A.L.S. Orozco, L.J.G. Villalba, Analysis of the GPS spoofing vulnerability in the drone 3DR solo, IEEE Access 7 (2019) 51782–51789.
[19] S.Z. Khan, M. Mohsin, W. Iqbal, On GPS spoofing of aerial platforms: a review of threats, challenges, methodologies, and future research directions, PeerJ Comput. Sci. 7 (2021) e507.
[20] B. Davidovich, B. Nassi, Y. Elovici, Towards the detection of GPS spoofing attacks against drones by analyzing camera's video stream, Sensors 22 (7) (2022) 2608.
[21] A. Jafarnia-Jahromi, A. Broumandan, J. Nielsen, G. Lachapelle, GPS vulnerability to spoofing threats and a review of antispoofing techniques, Int. J. Navig. Obs. 2012 (2012).
[22] D.P. Shepard, T.E. Humphreys, A.A. Fansler, Evaluation of the vulnerability of phasor measurement units to GPS spoofing attacks, Int. J. Crit. Infrastruct. Prot. 5 (3–4) (2012) 146–153.
[23] E. Ranyal, K. Jain, Unmanned aerial vehicle's vulnerability to gps spoofing a review, J. Indian Soc. Remote. Sens. 49 (3) (2021) 585–591.
[24] E. Schmidt, N. Gatsis, D. Akopian, A GPS spoofing detection and classification correlator-based technique using the LASSO, IEEE Trans. Aerosp. Electron. Syst. 56 (6) (2020) 4224–4237.
[25] A.Y. Javaid, F. Jahan, W. Sun, Analysis of global positioning system-based attacks and a novel global positioning system spoofing detection/mitigation algorithm for unmanned aerial vehicle simulation, Simulation 93 (5) (2017) 427–441.
[26] R. Skulstad, C. Syversen, M. Merz, N. Sokolova, T. Fossen, T. Johansen, Autonomous net recovery of fixed-wing UAV with single-frequency carrier-phase differential GNSS, IEEE Aerosp. Electron. Syst. Mag. 30 (5) (2015) 18–27.
[27] S. Ullah, K.H. Mohammadani, M.A. Khan, Z. Ren, R. Alkanhel, A. Muthanna, U. Tariq, Position-monitoring-based hybrid routing protocol for 3D UAV-based networks, Drones 6 (11) (2022) 327.
[28] I. Hussain, A. Anees, T.A. Al-Maadeed, M.T. Mustafa, Construction of s-box based on chaotic map and algebraic structures, Symmetry 11 (3) (2019) 351.
[29] A. Shafique, A noise-tolerant cryptosystem based on the decomposition of bit-planes and the analysis of chaotic gauss iterated map, Neural Comput. Appl. 34 (19) (2022) 16805–16828.
[30] I. Hussain, A. Anees, A.H. AlKhaldi, A. Algarni, M. Aslam, Construction of chaotic quantum magnets and matrix Lorenz systems S-boxes and their applications, Chinese J. Phys. 56 (4) (2018) 1609–1621.
[31] L. Xiao, P.-C. Ma, X.-M. Tang, G.-F. Sun, GNSS receiver anti-spoofing techniques: a review and future prospects, in: Electronics, Communications and Networks V: Proceedings of the 5th International Conference on Electronics, Communications and Networks, CECNet 2015, Springer, 2016, pp. 59–68.
[32] M. Spanghero, P. Papadimitratos, High-precision hardware oscillators ensemble for GNSS attack detection, in: 2022 IEEE Aerospace Conference, AERO, IEEE, 2022, pp. 1–11.
[33] S. Pullen, M. Joerger, GNSS integrity and receiver autonomous integrity monitoring (RAIM), in: Position, Navigation, and Timing Technologies in the 21st Century: Integrated Satellite Navigation, Sensor Systems, and Civil Applications, vol. 1, Wiley Online Library, 2020, pp. 591–617.
[34] B.A. Mahmood, D. Manivannan, Position based and hybrid routing protocols for mobile ad hoc networks: a survey, Wirel. Pers. Commun. 83 (2015) 1009–1033.
[35] M.R. Gholami, S. Gezici, E.G. Strom, Improved position estimation using hybrid TW-TOA and TDOA in cooperative networks, IEEE Trans. Signal Process. 60 (7) (2012) 3770–3785.
[36] C. Rani, H. Modares, R. Sriram, D. Mikulski, F.L. Lewis, Security of unmanned aerial vehicle systems against cyber-physical attacks, J. Déf. Model. Simul. 13 (3) (2016) 331–342.
[37] R.T. Ioannides, T. Pany, G. Gibbons, Known vulnerabilities of global navigation satellite systems, status, and potential mitigation techniques, Proc. IEEE 104 (6) (2016) 1174–1194.
[38] C. Günther, A survey of spoofing and counter-measures, NAVIGATION: J. Inst. Navig. 61 (3) (2014) 159–177.
[39] Y. Dang, C. Benzaïd, B. Yang, T. Taleb, Y. Shen, Deep-ensemble-learning-based GPS spoofing detection for cellular-connected UAVs, IEEE Internet Things J. 9 (24) (2022) 25068–25085.

[40] A. Anees, Y.-P.P. Chen, True real time pricing and combined power scheduling of electric appliances in residential energy management system, Appl. Energy 165 (2016) 592–600.

[41] M. Umer, I. Ashraf, Y. Park, et al., Enhanced machine learning ensemble approach for securing small unmanned aerial vehicles from GPS spoofing attacks, IEEE Access (2024).

[42] A. Anees, I. Hussain, U.M. Khokhar, F. Ahmed, S. Shaukat, Machine learning and applied cryptography, Secur. Commun. Netw. 2022 (2022) 1–3.

[43] M. Nayfeh, Y. Li, K. Al Shamaileh, V. Devabhaktuni, N. Kaabouch, Machine learning modeling of GPS features with applications to UAV location spoofing detection and classification, Comput. Secur. 126 (2023) 103085.

[44] R.A. Agyapong, M. Nabil, A.-R. Nuhu, M.I. Rasul, A. Homaifar, Efficient detection of gps spoofing attacks on unmanned aerial vehicles using deep learning, in: 2021 IEEE Symposium Series on Computational Intelligence, SSCI, IEEE, 2021, pp. 01–08.

[45] J. Pawlak, Y. Li, J. Price, M. Wright, K. Al Shamaileh, Q. Niyaz, V. Devabhaktuni, A machine learning approach for detecting and classifying jamming attacks against ofdm-based uavs, in: Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning, 2021, pp. 1–6.

[46] A. Iqbal, M.N. Aman, B. Sikdar, A deep learning based induced GNSS spoof detection framework, IEEE Trans. Mach. Learn. Commun. Netw. (2024).

[47] P. Borhani-Darian, H. Li, P. Wu, P. Closas, Detecting GNSS spoofing using deep learning, EURASIP J. Adv. Signal Process. 2024 (1) (2024) 14.

[48] M.M. Abrar, R. Islam, S. Satam, S. Shao, S. Hariri, P. Satam, GPS-IDS: An anomaly-based GPS spoofing attack detection framework for autonomous vehicles, 2024, arXiv preprint arXiv:2405.08359.

[49] Y. Dang, C. Benzaïd, B. Yang, T. Taleb, Deep learning for GPS spoofing detection in cellular-enabled UAV systems, in: 2021 International Conference on Networking and Network Applications, NaNA, IEEE, 2021, pp. 501–506.

[50] F. Tlili, S. Ayed, L.C. Fourati, A new hybrid adaptive deep learning-based framework for UAVs faults and attacks detection, IEEE Trans. Serv. Comput. (2023).

[51] J. Brewington, D. Kar, UAV gps spoofing detection via neural generative one-class classification, in: Proceedings of the Twenty-Fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing, 2023, pp. 492–497.

[52] X. Wei, C. Sun, X. Li, J. Ma, Machine Learning-Based Gnss Spoofing Detection for Uavs Using Doppler Frequency and Cn0, Available At SSRN 4769329.

[53] A. Aladi, E. Alsusa, UAV attack detection and mitigation using a localization verification-based autoencoder, IEEE Access (2023).

[54] K. Li, X. Lu, M. Akagi, M. Unoki, Contributions of Jitter and Shimmer in the voice for fake audio detection, IEEE Access 11 (2023) 84689–84698.

[55] F. Jalali-najafabadi, C. Gadepalli, D. Jarchi, B.M. Cheetham, Acoustic analysis and digital signal processing for the assessment of voice quality, Biomed. Signal Process. Control. 70 (2021) 103018.

[56] B. Li, S. Zhao, R. Zhang, L. Yang, Joint transmit power and trajectory optimization for two-way multihop UAV relaying networks, IEEE Internet Things J. 9 (23) (2022) 24417–24428.

[57] T.A. Al-Maadeed, I. Hussain, A. Anees, M.T. Mustafa, A image encryption algorithm based on chaotic Lorenz system and novel primitive polynomial S-boxes, Multimedia Tools Appl. 80 (2021) 24801–24822.

[58] F.S.D. Silva, E.P. Neto, H. Oliveira, D. Rosário, E. Cerqueira, C. Both, S. Zeadally, A.V. Neto, A survey on long-range wide-area network technology optimizations, IEEE Access 9 (2021) 106079–106106.

[59] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, Evaluation of the IoT LoRaWAN solution for distributed measurement applications, IEEE Trans. Instrum. Meas. 66 (12) (2017) 3340–3349.

[60] A. Shafique, A new algorithm for the construction of substitution box by using chaotic map, Eur. Phys. J. Plus 135 (2) (2020) 194.

[61] B. Bai, S. Nazir, Y. Bai, A. Anees, Security and provenance for Internet of Health Things: A systematic literature review, J. Softw.: Evol. Process. 33 (5) (2021) e2335.

[62] A. Shafique, J. Shahid, Novel image encryption cryptosystem based on binary bit planes extraction and multiple chaotic maps, Eur. Phys. J. Plus 133 (8) (2018) 331.

[63] T.S. Rappaport, Wireless Communications: Principles and Practice, Cambridge University Press, 2024.

[64] J. Frontera-Pons, F. Pascal, J.-P. Ovarlez, Adaptive nonzero-mean Gaussian detection, IEEE Trans. Geosci. Remote Sens. 55 (2) (2016) 1117–1124.

[65] D.K. Lee, J. In, S. Lee, Standard deviation and standard error of the mean, Korean J. Anesthesiol. 68 (3) (2015) 220.

[66] A. Anees, I. Hussain, A. Algarni, M. Aslam, et al., A robust watermarking scheme for online multimedia copyright protection using new chaotic map, Secur. Commun. Netw. 2018 (2018).

[67] Z. Zhang, Y. Li, X. He, L. Hsu, Resilient GNSS real-time kinematic precise positioning with inequality and equality constraints, GPS Solut. 27 (3) (2023) 116.

[68] I.L. Mallika, D.V. Ratnam, S. Raman, G. Sivavaraprasad, A new ionospheric model for single frequency GNSS user applications using Klobuchar model driven by auto regressive moving average (SAKARMA) method over Indian region, IEEE Access 8 (2020) 54535–54553.

[69] S. Jeong, GNSS spoofing detection using a maximum likelihood-based sliding window method, Plos One 15 (8) (2020) e0237146.

[70] K. Chang, Y. Yoo, J.-G. Baek, Anomaly detection using signal segmentation and one-class classification in diffusion process of semiconductor manufacturing, Sensors 21 (11) (2021) 3880.

[71] E. Shafiee, M.R. Mosavi, M. Moazedi, Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers, J. Navig. 71 (1) (2018) 169–188.

[72] S. Semanjski, I. Semanjski, W. De Wilde, A. Muls, Use of supervised machine learning for GNSS signal spoofing detection with validation on real-world meaconing and spoofing data—Part I, Sensors 20 (4) (2020) 1171.

[73] A. Shafique, A. Mehmood, M. Alawida, A.N. Khan, A.U.R. Khan, et al., A novel machine learning technique for selecting suitable image encryption algorithms for IoT applications, Wirel. Commun. Mob. Comput. 2022 (2022).

[74] D.G. Silva, L.C. Oliveira, M. Andrea, Jitter estimation algorithms for detection of pathological voices, EURASIP J. Adv. Signal Process. 2009 (2009) 1–9.

[75] J.P. Teixeira, C. Oliveira, C. Lopes, Vocal acoustic analysis–Jitter, Shimmer and hnr parameters, Procedia Technol. 9 (2013) 1112–1122.

[76] J.P. Teixeira, D. Ferreira, S.M. Carneiro, Análise acústica vocal-determinação do Jitter e Shimmer para diagnóstico de patalogias da fala, in: 6° Congresso Luso-MoÇambicano de Engenharia, 3° Congresso de Engenharia de MoÇambique, (6°) INEGI, 2011.

[77] T. Talaei Khoei, S. Ismail, N. Kaabouch, Dynamic selection techniques for detecting GPS spoofing attacks on UAVs, Sensors 22 (2) (2022) 662.

[78] H. Wang, H. Li, M. Zhong, M. Lu, A space-time-ambiguity decomposition method for DOA estimation enhancing anti-spoofing via rotating dual antennas, IEEE Trans. Aerosp. Electron. Syst. (2024).

[79] Y. Gao, J. Lian, B. Raj, R. Singh, Detection and evaluation of human and machine generated speech in spoofing attacks on automatic speaker verification systems, in: 2021 IEEE Spoken Language Technology Workshop, SLT, IEEE, 2021, pp. 544–551.

[80] E.A. Spazzapan, E.M.G. Fabbron, W.W.F. Toledo, A. de Souza Paula, V.C. de Castro Marino, Acoustic measurements through the life span: Jitter, Shimmer, and standard deviation of fundamental frequency, J. Voice (2024).

[81] G.I. Schuëller, H.A. Jensen, Computational methods in optimization considering uncertainties–an overview, Comput. Methods Appl. Mech. Engrg. 198 (1) (2008) 2–13.

[82] X. Shang, P. Ma, M. Yang, T. Chao, An efficient polynomial chaos-enhanced radial basis function approach for reliability-based design optimization, Struct. Multidiscip. Optim. 63 (2021) 789–805.

[83] A. Anees, Y.-P.P. Chen, Discriminative binary feature learning and quantization in biometric key generation, Pattern Recognit. 77 (2018) 289–305.

[84] P. Liu, K.-K.R. Choo, L. Wang, F. Huang, SVM or deep learning? A comparative study on remote sensing image classification, Soft Comput. 21 (2017) 7053–7065.

[85] I.W. Tsang, J.T. Kwok, P.-M. Cheung, N. Cristianini, Core vector machines: Fast SVM training on very large data sets, J. Mach. Learn. Res. 6 (4) (2005).

[86] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, A. Lopez, A comprehensive survey on support vector machine classification: Applications, challenges and trends, Neurocomputing 408 (2020) 189–215.

[87] S. Afifi, H. GholamHosseini, R. Sinha, FPGA implementations of SVM classifiers: A review, SN Comput. Sci. 1 (3) (2020) 133.

[88] B.-J. Huang, K.-Y. Lin, H. Wang, Millimeter-wave low power and miniature CMOS multicascode low-noise amplifiers with noise reduction topology, IEEE Trans. Microw. Theory Tech. 57 (12) (2009) 3049–3059.

[89] F. Bruccoleri, E.A. Klumperink, B. Nauta, Wide-band CMOS low-noise amplifier exploiting thermal noise canceling, IEEE J. Solid-State Circuits 39 (2) (2004) 275–282.

[90] G. Panice, S. Luongo, G. Gigante, D. Pascarella, C. Di Benedetto, A. Vozella, A. Pescapè, A SVM-based detection approach for GPS spoofing attacks to UAV, in: 2017 23rd International Conference on Automation and Computing, ICAC, IEEE, 2017, pp. 1–11.

[91] L. Meng, L. Yang, S. Ren, G. Tang, L. Zhang, F. Yang, W. Yang, An approach of linear regression-based UAV GPS spoofing detection, Wirel. Commun. Mob. Comput. 2021 (2021) 1–16.

[92] A. Shafique, A. Mehmood, M. Elhadef, Detecting signal spoofing attack in uavs using machine learning models, IEEE Access 9 (2021) 93803–93815.

[93] E.M. de Lima Pinto, R. Lachowski, M.E. Pellenz, M.C. Penna, R.D. Souza, A machine learning approach for detecting spoofing attacks in wireless sensor networks, in: 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications, AINA, IEEE, 2018, pp. 752–758.

[94] Q. Sun, X. Miao, Z. Guan, J. Wang, D. Gao, Spoofing attack detection using machine learning in cross-technology communication, Secur. Commun. Netw. 2021 (2021) 1–12.

[95] X. Ying, J. Mazer, G. Bernieri, M. Conti, L. Bushnell, R. Poovendran, Detecting ADS-B spoofing attacks using deep neural networks, in: 2019 IEEE Conference on Communications and Network Security, CNS, IEEE, 2019, pp. 187–195.