Deep Ensemble Learning based GPS Spoofing Detection for Cellular-Connected UAVs

Yongchao Dang, Chafika Benzaïd, Bin Yang, Tarik Taleb, *Senior Member, IEEE,* and Yulong Shen, *Member, IEEE*

***Abstract*—Unmanned Aerial Vehicles (UAVs) are an emerging technology in the 5G and beyond systems with the promise of assisting cellular communications and supporting IoT deployment in remote and density areas. Safe and secure navigation is essential for UAV remote and autonomous deployment. Indeed, the open-source simulator can use commercial software-defined radio tools to generate fake GPS signals and spoof the UAV GPS receiver to calculate wrong locations, deviating from the planned trajectory. Fortunately, the existing mobile positioning system can provide additional navigation for cellular-connected UAVs and verify the UAV GPS locations for spoofing detection, but it needs at least three base stations at the same time. In this paper, we propose a novel deep ensemble learning-based, mobile network-assisted UAV monitoring and tracking system for cellular-connected UAV spoofing detection. The proposed method uses path losses between base stations and UAVs communication to indicate the UAV trajectory deviation caused by GPS spoofing. To increase the detection accuracy, three statistics methods are adopted to remove environmental impacts on path losses. In addition, deep ensemble learning methods are deployed on the edge cloud servers and use the multi-layer perceptron (MLP) neural networks to analyze path losses statistical features for making a final decision, which has no additional requirements and energy consumption on UAVs. The experimental results show the effectiveness of our method in detecting GPS spoofing, achieving above 97% accuracy rate under two BSs, while it can still achieve at least 83% accuracy under only one BS.**

***Index Terms*—UAV, GPS spoofing, Multi-Layer Perceptron (MLP) , Deep ensemble learning, and Path loss.**

I. Introduction

UNMANNED Aerial Vehicles (UAVs) play the crucial role in the upcoming Internet of Things (IoT) platforms for remotely data gathering and aerially data transiting, supporting not only the capability of delivering IoT-based services but also the prospects of providing communications to dense and remote areas [1]. According to UAVs market report in [2], the value of UAV will reach 58.4B dollars by 2026. As more and more UAVs are coming, the security of autonomous UAVs must be addressed before the envisaged growth in UAV-based applications and services [3].

As a response, the Unmanned Aircraft Systems (UAS) Traffic Management (UTM) systems have been developed by the federal aviation administration for providing UAVs mission-related security services, including UAV authentication, flight plan authorization, real-time tracking and geofencing [4]. Reliable information on UAV position is essential for its operations and for a UTM system to carry out its mission. The global navigation satellite system, particularly Global Positioning System (GPS), is widely used to obtain UAV positioning information due to its global coverage and accuracy [4]. However, the unencrypted GPS signals are inherently vulnerable to spoofing attacks. In fact, an attacker can send counterfeit GPS signals to mislead the UAV's GPS receiver into generating false position information [5]. Furthermore, a malicious UAV may intentionally report forged GPS information to UTM system, resulting in violation of no-fly zone regulation and/or collision risks. Therefore, an effective GPS spoofing detection approach is vital to guarantee safe and secure integration of UAVs in the airspace.

GPS spoofing detection methods include GPS navigation signal analysis methods (e.g., [6]–[12]) and GPS navigation message encryption methods (e.g., [13]–[19]). The former signal analysis methods detect spoofing either through the difference of direction of arrival between the satellite signals and spoofer signals or through cross-correlation property between the military and civil GPS signals, while both need a ground-truth source in their detection processes. The latter encryption methods embed the authentication signature into navigation messages to guard against GPS spoofing attacks. However, the encryption processes need secured infrastructure support as well as more computing resources on the GPS receiver. The limited battery capacity and weight load of UAVs inhibit this GPS spoofing detection methods adoption in UAVs swarm systems.

Fortunately, the 3rd Generation Partnership Project (3GPP) has defined several standards to enhance long-term evolution support for Unmanned Aerial Systems (UAS) [20]–[23]. Those new standards allow the terrestrial cellular networks to provide identifying, locating, and tracking services for the UAS in order to enhance the security of the UAS operation. In this vein, the authors in [24] proposed a UAV tracking method, namely Adaptive Trustable Residence Area (ATRA), to detect the spoofed GPS position by leveraging up-link received signal strength indication. Regardless of the performance brought by

Corresponding author: Bin Yang.

Y. Dang is with the Department of Communications and Networking, School of Electrical Engineering, Aalto University, 02150 Espoo, Finland (e-mail: yongchao.dang@aalto.fi).

C. Benzaïd is with the Information Technology and Electrical Engineering, Oulu University, 90570 Oulu, Finland (e-mail: chafika.benzaid@oulu.fi).

B. Yang is with the School of Computer and Information Engineering, Chuzhou University, 239000 Anhui, China (e-mail: yangbinchi@gmail.com).

T. Taleb is with the Information Technology and Electrical Engineering, Oulu University, 90570 Oulu, Finland, and the Department of Computer and Information Security, Sejong University, Seoul, 05006 South Korea (e-mail: Tarik.Taleb@oulu.fi).

Y. Shen is with the School of Computer Science and Technology, Xidian University, 710071 Shaanxi, China (e-mail: ylshen@mail.xidian.edu.cn).

the ATRA method, it requires at least three base stations at the same time. To overcome the deficiency of the ATRA method, the authors in [25] introduced a deep neural network on the edge server that allows lively detecting GPS spoofing with three, two, or one base station, where the neural network model takes the statistical path losses features as inputs and produces the spoofing possibility as its output. Despite the acceptable performance provided by the neural network, it needs to collect the path losses data from different base stations which may lead to network congestion.

In this paper, we leverage the potential of deep ensemble methods and exploit the statistical features of path losses between a UAV and base stations to detect GPS spoofing for cellular-connected UAVs. Unlike the aforementioned GPS signal analysis and GPS information encryption contributions, the proposed approach is based on the analysis of the terrestrial mobile BS signal rather than GPS satellites signal from medium Earth orbit. Compared with the GPS satellites signal, the BS signal provides immunity from ionospheric interference, which makes it more stable and reliable for detecting the UAV trajectory deviation caused by GPS spoofing. In addition, our approach needs no additional hardware or computation load at the UAV. In fact, the detection task is accomplished at the Multi-access Edge Computing (MEC) servers and edge cloud server based on the path losses received from the mobile network. Moreover, its effectiveness is insensitive to the changes of the environmental conditions, thanks to the use of statistical properties of path losses. Finally, the use of deep ensemble learning amongst edge servers help to make collaborations between BSs, which can mitigate the down-tilt directional sector antennas impacts on spoofing detection performance. The main contributions of this paper are summarized as follows:

- We first formulate the spoofing detection processes as a nonlinear and non-convex optimization problem, which is subject to the threshold, the statistical feature weights, and the number of BSs. The goal of the optimization function is to minimize the sum of hypothesis test errors of GPS spoofing by optimizing the threshold of the hypothesis test and the statistical feature weights of the differences between the theoretical path losses and the actual ones provided by BSs.
- Due to the path losses vulnerable to environmental changes, we adopt three statistical methods to analyze the statistical properties of path losses, including moments, quartile, and probability distributions, for making our spoofing detection insensitive to the changes in environmental conditions.
- We further propose a MEC architecture for deep ensemble learning-based GPS spoofing detection. In MEC servers, each MLP works independently to predict the spoofing probability of a GPS position reported by each BS. To decide whether the GPS position is spoofed or not, six types of deep learning models are deployed into the edge cloud server to integrate MLPs' individual prediction results.
- In the end, we build a simulation platform using the 3GPP defined path losses model and the TensorFlow deep learning models. The experimental results illustrate the effectiveness of our proposed approach in detecting the spoofed GPS positions even with only one BS.

TABLE I
LIST OF ABBREVIATIONS USED.

Abbreviations	Definitions
ATRA	Adaptive Trustable Residence Area
BS	Base Station
DoA	Direction of Arrival
DT	Decision Tree
GNB	Gaussian Naive Bayes
GPS	Global Positioning System
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
LoS	Line of Sight
LR	Logistic Regression
MEC	Multi-access Edge Computing
ML	Machine Learning
MLP	Multi-Layer Perceptron
MPS	Mobile Positioning System
MVSK	Mean Variance Skewnee Kurtosis
NLoS	None-Line of Sight
OCCS	Operator Command and Control Service
PL	Path Loss
RSS	Received Signal Strength
SDPS	Supplementary Data Provider Service
SVMK	Support Vector Machine Kernel
SVMG	Support Vector Machine Gamma
UAV	Unmanned Aerial Vehicle
UAS	Unmanned Aerial System
UFC	UAV Flight Controller
UTM	Unmanned aircraft systems Traffic Management
WD	Wasserstein Distance
WP	Wat Point
3GPP	3rd Generation Partnership Project

This manuscript contains a number of abbreviations. For the sake of ease of readership, Table I lists the abbreviations used in this paper. The rest of the paper is organized as follows. Section II summarizes related work in the literature. The system model, path loss model and hypothesis testing are described in Section III. Section IV introduces three statistical methods and formulates the GPS spoofing detection as an optimization problem. A MLP-based ensemble approach is proposed in Section V. Experiment results are illustrated in Section VI. Section VII concludes the paper.

II. RELATED WORK

Over the last decade, much research efforts have been dedicated to studying the GPS spoofing problem. The proposed contributions can be categorized into the following five classes.

A. Navigation Signal Analysis Approaches

GPS navigation signal analysis is widely used to determine whether the GPS is spoofed or not. The authors in [9] proposed a multi-antenna spoofing detection technique which analyzes the DoA of GPS signals to discriminate between authentic and fake GPS signals. The received signals are considered spoofed if they arrive from the same source in space. Similarly, the work in [10] introduces a spatial signal processing approach for GPS spoofing detection and mitigation. The approach leverages multi-antenna reception and null-steering for, respectively, identifying and filtering out fake GPS signals. The adoption of the aforementioned approaches requires multiple antennas and incurs more computational load on the GPS receiver.

The methods in [11] and [12] rely on the cross-correlation between encrypted/military GPS signals received by a trusted receiver and the defended civil receiver in order to detect the spoofing of unencrypted GPS signals. A low cross-correlation indicates the presence of a spoofing attack. While these methods do not depend on a multi-antenna GPS receiver, they require a communication link between the defended receiver and a secure receiver to perform the cross-correlation.

B. Cryptal-protected Navigation Message Approaches

To protect civil GPS receivers against spoofing attacks, the approaches falling in this category consider either the encryption or the authentication of the navigation messages. The navigation message encryption methods (e.g., [16]) ensure the confidentiality of the navigation message by ciphering its content. The navigation message encryption methods are deemed impractical as their implementation requires changes to the GPS interface specifications [17]. The navigation message authentication techniques aim at guaranteeing the integrity of the navigation message and the authenticity of its source by digitally signing its content. For instance, Wesson *et al.* [18] combined signature-based authentication of GPS navigation messages with a statistical hypothesis test to prevent counterfeit navigation messages. Wu *et al.* [14] leveraged SM cryptographic algorithms to authenticate the BeiDou-II navigation messages. Similarly, the work in [13] presented a BeiDou-II navigation message authentication scheme based on digital signatures generated by an elliptic curve digital signature algorithm. In [19], a trusted execution environment was used to generate cryptographically-signed GPS messages in order to prevent their forgery. The European Galileo global navigation satellite system will provide an open navigation message authentication service (OS-NMA) [26] based on an adaptation of the Timed Efficient Stream Loss-tolerant Authentication (TESLA) scheme. OS-NMA enables source authentication and message integrity by transmitting the navigation message along with a message authentication code. The message authentication code key is derived from a one-way hash chain and is released after a pre-defined period of time. Even though navigation message authentication techniques are considered a practical and effective defense against GPS spoofing attack, their use induces significant computational cost and latency due to signature verification. Moreover, some of them (e.g., OS-NMA) require the availability of loose time synchronization [27]. Finally, navigation message authentication based methods cannot withstand replay attacks.

C. Inertial Navigation System based Approaches

An INS uses an IMU comprising various inertial sensors (e.g., accelerometers, gyroscopes, and magnetometers) to assist navigation based on sensor measurements. The INS-based approaches detect GPS spoofing attacks by using the position estimated from the IMU readings to cross-validate the veracity of the reported GPS position. Lee *et al.* [28] used a fixed probability of false alarms and the root mean square error between the accelerometer outputs and the acceleration estimated from the GP outputs to detect spoofing GPS signals. In [29], the accelerometer readings were leveraged to identify spoofing GPS signals, where the spoofing decision is based on the probability density function. Feng *et al.* [30] used on-board gyroscopes' measurements to determine whether a UAV has been hijacked by GPS spoofing.

The INS has the advantage to operate without reliance on any external signals, making it immune to spoofing attacks. However, the main issue with INS is the error accumulation of the IMU measurements over time, which can negatively impact the detection accuracy.

D. Mobile Cellular Network based Approaches

Recently, a new class of anti-GPS spoofing approaches has been introduced leveraging the localization ability of mobile cellular networks to relocate the target and discriminate the spoofed GPS positions in the base stations' coverage area. The work in [31] exploited the strength of signals received from base stations of a 2G cellular network to estimate the vehicle position. The estimated position is then used to cross-check the consistency of the vehicle's GPS position. The authors in [32] used the position estimates obtained by the data relative to the neighboring cells in order to check the validity of the GPS position of a smartphone. Different from the aforementioned solutions, the work in [33] considered the use of information received from 5G network to recognize spoofed UAV's GPS positions reported to UTM. The proposed approach utilizes the Received Signal Strength (RSS) collected from three base stations to infer the trust area within which the GPS position should be located in order to be considered genuine.

E. Machine Learning based Approaches

The capability of ML to learn from historical data to unveil hidden patterns has driven the recent trend of using ML techniques for GPS spoofing detection in UAV environments. In this vein, different ML methods have been proposed to detect GPS spoofing either by classifying the spoofed GPS signal directly (e.g., [34]–[39]) or by verifying the GPS information supplementally (e.g., [40]–[43]). The GPS signal classification methods take advantage of the ML classifier techniques to discriminate the fake GPS signal from the actual GPS signal, while the GPS information verification methods make use of

TABLE II
SUMMARY OF DETECTION METHODS FOR GPS SPOOFING ATTACK.

Work	Detection method	Requirements				
		Data source	Computing location	Hardware	Energy	Loads
[6]–[12]	Signal analysis	GPS signals	GPS receiver	Multi-receivers	UAV	Yes
[34]–[39]	ML-based signal analysis					
[13]–[19]	Encryption	GPS message	GPS receiver	Secure hardware	UAV	No
[28]–[30]	Internal sensor	Sensor data	UAV board	Different sensors	UAV	Yes
[40]–[43]	ML-based GPS verification					
[31]–[33]	Mobile positioning system	Cellular signals	Edge server	Multi-BSs	BS	No
Ours	Deep ensemble learning	Path losses	Edge server	Single/multi-BSs	BS	No

the ML-based location recognition techniques to prove the authenticity of the GPS locations.

1) ML-based GPS signal classification: Unlike the navigation signal analysis approaches discussed above, this method relies on ML classification models to analyze the GPS navigation signal characteristics for separating fake signals from the authentic ones.

Manesh *et al.* [34] exploited the received GPS signal characteristics, such as pseudo range, Doppler shift and signal-to-noise ratio, to build a supervised neural network model for detecting GPS spoofing against unmanned aerial systems. Similarly, the spoofing detection approach in [35] used a neural network to analyze the differences between the spoofing and authentic signal patterns. The neural network takes as inputs the delta criterion, coefficient of early and late phase criterion, and total levels of signal. The authors in [36] proposed a GPS spoofing detection approach that leverages a supervised ML algorithm, specifically Support Vector Machine (SVM), to analyze the cross-correlation among the GPS signals of multiple GPS receivers. The work in [37] introduced RBF SVM, Ada Boost, Decision Trees, K-Nearest Neighbors, Random Forests methods for GPS signal's radio frequency interference analysis under the assumption that the attacker was not able to null the satellite signal. It is worth mentioning that the space weather changes diminish the features of raw GPS signals [38] and the attacker can broadcast the same frequency jamming GPS signals for blinding the GPS receiver [39], which can considerably hinder the effectiveness of the aforementioned approaches.

2) ML-based GPS information verification: To overcome the weakness of GPS signal analyses based spoofing detection methods, the external positioning techniques, such as INS and MPS, are also used to not only provide location services for UAVs when the GPS is unavailable but also to cross-validate the authenticity of GPS information. Recently, ML techniques have been leveraged in both INS and MPS based GPS spoofing detection methods for improving the detection accuracy.

Motivated by the fact that a camera view, one of the INS components, endows a UAV with the ability to locate itself and recognize its surroundings, a number of studies have developed UAV camera view based ML methods to detect GPS spoofing. For instance, the authors in [40] used a hidden Markov model to model the human operator approaches in detecting fake GPS locations based on analysis of aerial photos taken by the UAV. Based on the PoseNet convolutional neural networks, the work in [41] used aerial images to provide the UAV's GPS position. In the same vein, the deep convolutional neural networks was adopted in [42] to enable the UAV navigation in GPS-denied environments using only visual information. Nevertheless, the acquisition, transmission and processing of camera views are time consuming operations which can introduce noticeable latency for detecting the GPS spoofing.

As aforementioned, MPS has been used to locate and monitor the mobile targets using at least three BSs or multi-antennas for acquiring an accurate target position. However, some airspace areas may be covered by less than three BSs and even only one BS, which can limit the effectiveness of existing MPS based location tracking and spoofing detection solutions. Luckily, ML is a potential enabler for building MPS-based spoofing detection services that can perform well even in some worse cases. In [43], Xiao *et al.* investigated the potential of recurrent neural networks in recognizing the deviation in the UAV's trajectory caused GPS spoofing. To this end, the DoA measurements of BSs' signals from the UAV equipment are analyzed by the proposed recurrent neural network models. Nevertheless, the adoption of their approach requires UAVs to be equipped with cylindrical antenna arrays and incurs more computational load on the UAVs to estimate the DoA values.

Table.II summarizes the aforementioned spoofing detection methods based on the requirements of data source, computing location, hardware, energy, and UAV loads. Based on the difference in data sources, the spoofing detection methods can be divided into four categories, including GPS signals analysis methods, GPS message encryption methods, internal sensors data analysis methods, and cellular signals analysis methods. Both signal analysis or ML-based signal analysis methods require additional computing resources on the GPS receiver and rely on multi-GPS receivers proving either signals directions or a secure template, which increases the UAV energy consumption and brings extra loads on the UAV. The GPS message encryption methods execute encryption and decryption processes on the GPS receiver secure hardware, which consumes more energy but without the increase in the UAV loads. The sensor data-based spoofing detection

methods detect spoofing through the differences between GPS location and internal sensors' location, which needs support from UAV board computer and different kinds of sensors. However, the internal sensor and ML-based GPS verification methods consume UAV energy for location computation and put more loads on the UAV for carrying sensors. Although the MPS-based spoofing detection methods use the cellular signals and are deployed on the edge servers without energy consummation and load requirements on the UAV, they require at least three BSs at the same time for triangular localization.

Despite the merits of the above mentioned solutions, an effective approach to detect GPS spoofing while accommodating to resource, cost and environmental constraints of a UAV environment is still missing. To fill this gap, we leverage the potential of ML and exploit the statistical features of path losses between UAV and 5G base stations (BSs) to devise an effective 5G-assisted GPS spoofing detection approach. The proposed approach needs no additional hardware or computation and load at the UAV. Moreover, its effectiveness is less prone to changes in the environmental conditions, thanks to the stability introduced by the statistical features. Furthermore, we use deep learning method to detect GPS spoofing with one single BS and also can achieve cooperation between different BSs with the help of ensemble learning. By using the path losses that can be obtained from the BSs broadly and speedily, and taking advantage of the capability of ML to deliver faster decisions, the proposed approach will empower live detection of spoofed GPS positions.

III. System Model and Performance Metrics

A. Network Model

As illustrated in Fig. 1, we consider a space-aerial-terrestrial integrated network consisting of a space GPS satellites network providing location service for UAVs, an aerial vehicle network including a target UAV u and an aerial GPS spoofer, and a terrestrial mobile network with N BSs connecting with aerial vehicles through wireless channels.

In the aerial network, the aerial GPS spoofer sends fake GPS signal to mislead the target u into deviating from its planned trajectory. As shown in Fig. 1, if there is no GPS spoofing, u will reach the waypoint P_u of the planned trajectory starting from the waypoint P_j to the waypoint P_{j+1}. Once GPS spoofed, it will reach the waypoint $P_u^{'}$ of the spoofed trajectory starting from P_j to the waypoint $P_{j+1}^{'}$.

The terrestrial mobile network is mainly used for GPS spoofing detection. The terrestrial BSs first collect the PL data during the communications between u and these BSs. Three kinds of statistical methods, introduced in the next section, are then applied to process the PL data for reducing the negative impact of the environmental conditions on it. In addition, the edge servers deployed near the BSs provide computation resources in order to reduce the data transmission latency and improve spoofing detection effectiveness. Finally, the edge servers receives the processed PL data through the core network connecting these BSs, and further decides whether the GPS position is spoofed or not. The notations uses in this paper are summarized in Table III for clarity.

TABLE III
Notations used in the paper.

Symbol	Definitions
u	Target UAV
N	The number of base stations
P	GPS position, $P = (\text{Lat}_u, \text{Lon}_u, \text{Alt}_u)$
$\mathcal{P}$	Planned way points
$\mathcal{R}$	Reported way points
$\lambda/\widetilde{\lambda}$	The probability of LoS/NLoS link
$L/\overline{L}$	The actual/theoretical path loss
ΔL	The difference between actual and theoretical path loss.
$L^{'}$	The reference path loss in a free space
$d_{iu}/d_{iu}^{'}$	The 2D/3D distance between i^{th} BS and u
h_u	The altitude of u
T	The threshold of hypothesis testing
E	The number of data points in a time slot
f_c	The carrier frequency 2 GHz
$\mathcal{F}$	The statistical features set
F	The ensemble statistical features from one BS
$\mathbb{F}$	The weighted sum of F from different BSs
$C(\cdot)$	The cumulative distribution function
$\mathbb{C}$	The set of MLP neural networks
ω	The weight of statistical feature
Ω	The weight of BS ensemble feature
B	The binary cross-entropy loss
γ	Fraction to create bootstrapped training data
dE	The system GPS error tolerance
α	The importance coefficient for miss detection
$\mathfrak{M}$	Ensemble ML methods
$\mathfrak{D}$	Data set of ML predictions
Θ	Thresholds for ML ensemble

B. Path Loss Model

The aerial UAV and terrestrial BS communication contains both LoS and Non-LoS (NLoS) links. The following air-to-ground path loss model for UAV u and BS i has been tested by the 3GPP in [21], and is determined as

$$\overline{L}_{iu} = \lambda_{iu} \times L_{iu}^{\mathfrak{L}} + \widetilde{\lambda}_{iu} \times L_{iu}^{\mathfrak{N}}, \tag{1}$$

where $L_{iu}^{\mathfrak{L}}$ and $L_{iu}^{\mathfrak{N}}$ denote the path loss of LoS link and NLoS link, respectively. λ_{iu} is the probability of LoS link and $\widetilde{\lambda}_{iu}$ is the probability of NLoS link, and $\widetilde{\lambda}_{iu} = 1 - \lambda_{iu}$.

For an urban micro UAV, the LoS probability, λ_{iu}, is given by

$$\lambda_{iu} = \begin{cases} 1 & \text{if } d_{iu} \leq d_1 \\ \frac{d_1}{d_{iu}} + \exp(\frac{-d_{iu}}{q_1})(1 - \frac{d_1}{d_{iu}}) & \text{if } d_{iu} > d_1 \end{cases} \tag{2}$$

where $d_1 = \max(294.05 \log_{10}(h_u) - 432.94, 18)$ and $q_1 = 233.98 \log_{10}(h_u) - 0.95$. d_{iu} is the horizontal distance between

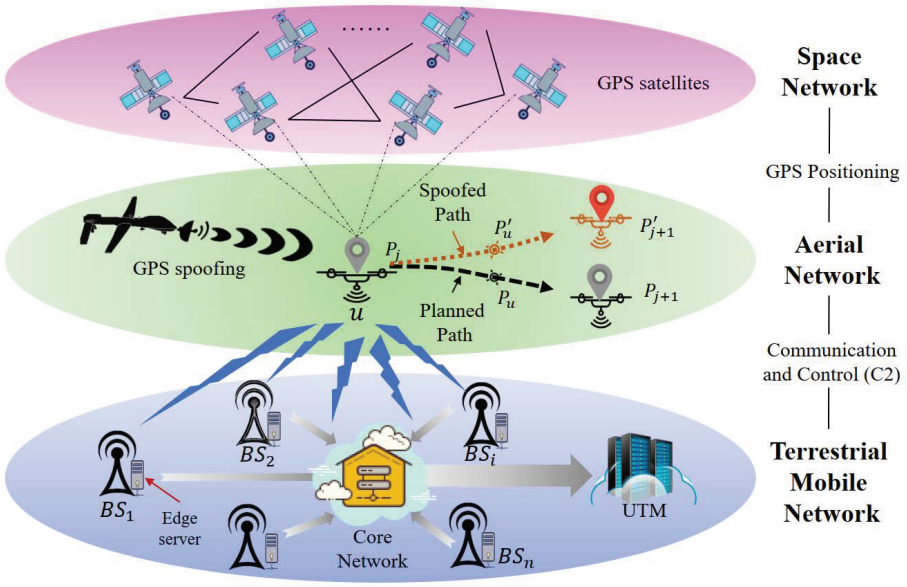


Fig. 1. The space-aerial-terrestrial integrated network model.

i^{th} BS and u, and h_u is the u's altitude, as shown in Fig.2. Note that h_u can be larger, equal, or smaller than the BS_i's height.

The path losses $L_{iu}^{\mathfrak{L}}$ and $L_{iu}^{\mathfrak{N}}$ are expressed as:

$$L_{iu}^{\mathfrak{L}} = \max\{L', 30.9 + (22.25 - 0.5\log_{10}(h_u)) \times \log_{10}(d_{iu}^{'}) + 20\log_{10}(f_c)\}, \quad (3)$$

and

$$L_{iu}^{\mathfrak{N}} = \max\{L_{iu}^{\mathfrak{L}}, 32.4 + (43.2 - 7.6\log_{10}(h_u)) \times \log_{10}(d_{iu}^{'}) + 20\log_{10}(f_c)\}, \quad (4)$$

where the carrier frequency f_c is equal to 2 GHz. $L' = 15.3 + 37.6\log_{10}(d_{iu}^{'})$ is the path loss for a carrier frequency of 2 GHZ in a free space, and $d_{iu}^{'}$ is the distance between BS_i and u, as shown in Fig. 2. P_u is the position of u and P_i is the position of the i^{th} BS. Alternatively, the path loss model above can be further extended into the Rician channel that includes both large scale and small scale fading, following procedures in [21]. It is worth mentioning that the Rician fading impacts are insignificant thanks to the use of statistical methods on the path losses.

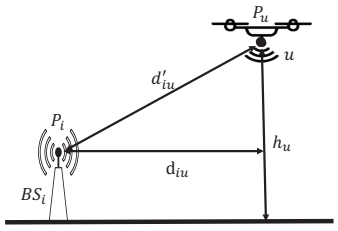


Fig. 2. The illustration of the parameters d_{iu}, $d_{iu}^{'}$ and h_u.

It is notable that there are three unknown parameters in (2), (3) and (4), i.e., the UAV height, the horizontal distance between the target UAV and each BS, and the distance between them. To obtain these three parameters, the edge server need to know the locations of the UAV and BS, which are obtained according to the Federal Aviation Administration (FAA) regulation [44]. Under such a regulation, the target UAV needs to broadcast its location for safety and security purposes [44]. The edge server can also ask the location of each BS from the mobile network operator. Based on the locations of the UAV and BS, the edge server can obtain these three parameters and also determine the theoretical path loss between the target UAV and BS using the path loss model of (2), (3) and (4). Furthermore, the edge server can obtain the actual path loss based on the UAV transmit power and the received signal strength at BS. This is because the edge server can control the UAV transmission power through an edge UAV flight controller [45] and also can get the received signal strength through mobile network services defined by 3GPP [21].

C. Hypothesis Testing

Based on the observation that different UAV's positions generally result in different values of path loss between the BS and the UAV, UTM uses the difference between the actual path loss L_{iu} and the theoretical one $\overline{L}_{iu}$ to decide whether the GPS position of UAV u is spoofed or not. Here, the actual path loss and the theoretical path loss are determined according to Subsection III-B. Thus, we have

$$\Delta L_{iu} = \left| L_{iu} - \overline{L}_{iu} \right|, \quad (5)$$

where ΔL_{iu} denotes the absolute value of the difference between the actual path loss and the theoretical one. Notice that an actual GPS position of a UAV corresponds to a theoretical path loss nearly the same as the actual one. Meanwhile, a spoofed GPS position corresponds to a theoretical path loss deviating from the actual one, which means that a bigger ΔL_{iu}

indicates a higher probability that the GPS position of the UAV is spoofed. Therefore, the GPS spoofing detection problem can be formulated as a threshold-based hypothesis testing given by

$$\begin{cases} H_0: & \Delta L_{iu} > T, \\ H_1: & \Delta L_{iu} \leq T, \end{cases} \tag{6}$$

where T denotes a threshold of the hypothesis testing. The null hypothesis H_0 represents that the GPS position is spoofed. H_0 is accepted if ΔL_{iu} is above the threshold T. On the other hand, a true alternative hypothesis H_1 means that there is no GPS spoofing.

Although the distance between UAV and BS is the main factor affecting the path loss in the threshold based hypothesis testing, other environmental factors (e.g., cloud, temperate and vapor) can also impact the path loss, which may lead to a wrong decision on the GPS spoofing. Accordingly, the threshold based hypothesis testing for GPS spoofing detection faces the following significant challenges. Firstly, the path loss of a data transmission is more likely to be affected by the environment, which may result in increased spoofing detection errors. Secondly, the threshold value has a noticeable impact on the accuracy of the hypothesis testing at different time slots. Indeed, a bigger threshold value could lead to a higher probability of miss detection, while a smaller threshold value could result in a higher probability of false alarms. Thus, determining the appropriate threshold value is crucial, yet a difficult task. Thirdly, the hypothesis testing results reported by different BSs should be given different weights to determine the final decision. The rationale behind assigning different weights is that a larger distance between a BS and a UAV could lead to a higher error of hypothesis testing result.

To address the aforementioned challenges, we leverage the potential of both statistical methods and ML to devise an effective GPS spoofing detection approach. Three statistical methods are used to extract the statistical properties of path losses of multiple data transmissions, which allows to remove the effects caused by the changing environmental conditions. To deal with the threshold and weight setting issues, we formulate them as a constrained optimization function and propose a MLP-based ensemble approach to solve it.

IV. Problem Formulation

In this section, we first introduce three typical statistical methods to analyze the PL data and eliminate the negative effect of environment on the spoofing detection. Based on these PL data, we further formulate the spoofing detection as a constrained optimization problem.

A. Statistical Methods

The following three statistical methods are provided to calculate the statistical metrics of path losses of multiple data transmissions.

1) Mean-Variance-Skewness-Kurtosis (MVSK): **MVSK** involves important numerical characteristics into the analysis of a path loss, including the Mean, the second central moment Variance, the third standardized moment Skewness, and the fourth standardized moment Kurtosis [46]. To obtain the MVSK metrics of path losses, we assume that the number of data transmissions that occur in a time interval t is E. The corresponding actual path loss set $L_{iu}(t)$ is expressed as:

$$L_{iu}(t) = \{L_{iu}^1(t), ..., L_{iu}^e(t), ..., L_{iu}^E(t)\}, \tag{7}$$

where $L_{iu}^e(t)$ denotes the e^{th} path loss in the time interval t. Similarly, the theoretical path loss set $\overline{L}_{iu}(t)$ is defined as:

$$\overline{L}_{iu}(t) = \{\overline{L}_{iu}^1(t), ..., \overline{L}_{iu}^e(t), ..., \overline{L}_{iu}^E(t)\}, \tag{8}$$

where $\overline{L}_{iu}^e(t)$ denotes the e^{th} theoretical path loss in the time interval t. According to the MVSK method, we have

$$\begin{cases} L_{iu}^M(t) = \frac{1}{E}\sum\limits_{e=1}^{E} L_{iu}^e(t), \\ L_{iu}^V(t) = \frac{1}{E}\sum\limits_{e=1}^{E} (L_{iu}^e(t) - L_{iu}^M(t))^2, \\ L_{iu}^S(t) = \frac{1}{E}\sum\limits_{e=1}^{E} (\frac{L_{iu}^e(t) - L_{iu}^M(t)}{\sqrt{L_{iu}^V(t)}})^3, \\ L_{iu}^K(t) = \frac{1}{E}\sum\limits_{e=1}^{E} (\frac{L_{iu}^e(t) - L_{iu}^M(t)}{\sqrt{L_{iu}^V(t)}})^4, \end{cases} \tag{9}$$

where $L_{iu}^M(t)$, $L_{iu}^V(t)$, $L_{iu}^S(t)$ and $L_{iu}^K(t)$ represent, respectively, the mean, variance, skewness, and kurtosis values of actual path losses in time interval t. Similarly, we can obtain the theoretical ones $\overline{L}_{iu}^M(t)$, $\overline{L}_{iu}^V(t)$, $\overline{L}_{iu}^S(t)$ and $\overline{L}_{iu}^K(t)$.

The difference $\Delta L_{iu}^x(t)$ between the actual and theoretical MVSK metrics is expressed as

$$\Delta L_{iu}^x(t) = \left| L_{iu}^x(t) - \overline{L}_{iu}^x(t) \right|, \tag{10}$$

where $x \in \{M, V, S, K\}$. For each difference $\Delta L_{iu}^x(t)$, the threshold introduced in the hypothesis testing in (6) is denoted as $T_{iu}^x(t)$.

2) BOX: **BOX** is a method of descriptive statistics including the minimum ($Q0$), the first quartiles ($Q1$), the sample median ($Q2$), the third quartiles ($Q3$) and the maximum ($Q4$) of the path loss set $L_{iu}(t)$ [47]. $L_{iu}^{Q0}(t)$ and $L_{iu}^{Q4}(t)$ denote, respectively, the lowest and the largest path loss excluding any outliers. $L_{iu}^{Q2}(t)$ and $L_{iu}^{Q3}(t)$ represent the median and the median of the upper half of the $L_{iu}(t)$, respectively. Here, outliers represent the observations that fall below $L_{iu}^{Q1}(t) - 1.5 * \text{IQR}$ or above $L_{iu}^{Q3}(t) + 1.5 * \text{IQR}$, where IQR is the Inter-Quartile Range and is equal to $L_{iu}^{Q3}(t) - L_{iu}^{Q1}(t)$.

Similarly, the BOX descriptive statistics of theoretical path losses are denoted as $\overline{L}_{iu}^{Q0}(t)$, $\overline{L}_{iu}^{Q1}(t)$, $\overline{L}_{iu}^{Q2}(t)$, $\overline{L}_{iu}^{Q3}(t)$ and $\overline{L}_{iu}^{Q4}(t)$. Using (10), we can obtain $\Delta L_{iu}^x(t)$; the x quartiles difference between the actual and theoretical values, where $x \in \{Q0, Q1, Q2,\ Q3, Q4\}$. The corresponding threshold is denoted as $T_{iu}^x(t)$ in the hypothesis testing for GPS spoofing detection.

3) Wasserstein Distance (WD): **WD** is a metric to estimate the distance between two probability distributions [48]. We use $\Delta L_{iu}^W(t)$ to denote the WD distance between the actual path loss $L_{iu}(t)$ and the theoretical one $\overline{L}_{iu}(t)$. Then, we have

$$\Delta L_{iu}^{W}(t) = \left(\int_0^1 \left| C_{L_{iu}(t)}^{-1}(u) - \overline{C}_{\overline{L}_{iu}(t)}^{-1}(u) \right|^2 du\right)^{\frac{1}{2}}, \quad (11)$$

where $C_{L_{iu}(t)}^{-1}$ and $C_{\overline{L}_{iu}(t)}^{-1}$ are the corresponding inverse cumulative distribution functions (CDFs) of $L_{iu}(t)$ and $\overline{L}_{iu}(t)$, respectively. The WD distance $\Delta L_{iu}^{W}(t)$ can also be used to detect the GPS spoofing according to the hypothesis testing with the threshold $T_{iu}^{W}(t)$.

B. Optimization Problem

We use $F_{iu}(t)$ to denote the weighted statistical feature that represents the authenticity of the PL data provided by the i^{th} BS at time t. Since the PL differences under different statistical methods have different influences on the spoofing hypothesis testing results, we can define $F_{iu}(t)$ as the weighted sum of PL differences $\Delta L_{iu}^{x}(t)$, which is expressed as

$$F_{iu}(t) = \sum_{x \in \mathcal{F}} \omega_{iu}^{x} \times \Delta L_{iu}^{x}(t), \quad (12)$$

where $\omega_{iu}^{x}$is the weight for the statistical feature x and $\mathcal{F} = \{M, V, S, K, Q0, Q1, Q2, Q3, Q4, W\}$ is the set of statistical features.

At time t, u broadcasts signals to its nearby n BSs located at different positions. We know that a higher distance between u and a BS can lead to a bigger error of PL value [33], and thus it further incurs a bigger error of the ensemble statistical features. We use $\mathbb{F}_u(t)$ to denote the ensemble feature of the n BSs at time t. To reduce the ensemble error, we define $\mathbb{F}_u(t)$ as the weighted sum of $F_{iu}(t)$; i.e.,

$$\mathbb{F}_u(t) = \sum_{i=1}^{n} \Omega_{iu} \times F_{iu}(t), \quad (13)$$

where Ω_{iu} represents the weight of the i^{th} BS ensemble feature.

The threshold-based hypothesis testing given in (6) can be rewritten as

$$\begin{cases} H_0: & \mathbb{F}_u(t) > T_u(t), \\ H_1: & \mathbb{F}_u(t) \leq T_u(t), \end{cases} \quad (14)$$

where $T_u(t)$ is the threshold of hypothesis testing.

To ensure the spoofing detection performance at UTM, our objective is to minimize the total errors of the hypothesis testing. This can be formulated as the following optimization problem.

$$\min_{\omega, \Omega, T} \quad \sum_{i=1}^{n} \sum_{x \in \mathcal{F}} \phi(\omega, \Omega, T), \quad (15)$$

$$\begin{aligned} \text{Subject to:} \quad & 0 \leqslant \omega \leqslant 1, && (15a) \\ & 0 \leqslant \Omega \leqslant 1, && (15b) \\ & 1 \leqslant n \leqslant N, && (15c) \\ & \phi(\cdot) \in \{0, \alpha, \beta\}, && (15d) \end{aligned}$$

where $\phi(\cdot)$ is an indicator function of ω, Ω and T to measure the error of the hypothesis testing. $\omega = \{\omega_{iu}^{x} | x \in \mathcal{F}, i \in [1, n]\}$, $\Omega = \{\Omega_{iu} | i \in [1, n]\}$, and $T = T_u(t)$. $\phi(\cdot) = \alpha$ is used to measure the missed detection, i.e., if H_0 is true but UTM approves H_1. $\phi(\cdot) = \beta$ is used to measure the false alarm, i.e., if H_1 is true but UTM approves H_0. Otherwise, $\phi(\cdot) = 0$. Here, $\alpha > \beta$, which indicates that the missed detection is much more harmful than the false alarm because the former can cause violation of no-fly zone regulation and collision risks. Meanwhile, $\alpha + \beta = 1$, which is to let the objective function of Eq. (15) falls within a limited range. Otherwise, $\phi(\cdot) = 0$. Constraints (15a) and (15b) denote the range of the weights ω_{iu}^{x} and Ω_{iu}, respectively. Constraint (15c) denotes the range of the number of BSs, where $N = 9$ according to the 3GPP trials results in [21]. Constraint (15d) denotes that $\phi(\cdot)$ is equal to 0, α or β.

Note that this is a nonconvex and nonlinear optimization problem, which is generally challenging to be solved. ML, particularly deep learning, is a promising direction to deal with this challenging problem, owing to its ability of solving complex problems, while achieving the desired performance/complexity balance. Thus, we propose a MLP-based ensemble approach to solve the optimization problem in the following section.

V. MLP-based Ensemble Approach

This section elaborates on the structure and algorithms used to build the multi-MLP ensemble model for GPS spoofing detection.

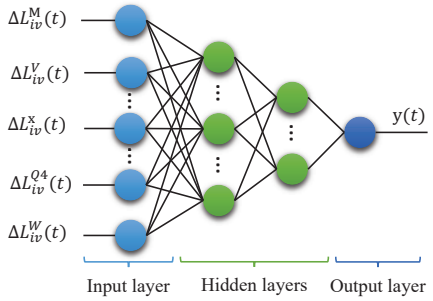


Fig. 3. The structure of the MLP.

A. MLP Model

As shown in Fig. 3, the MLP is a deep learning neural network consisting of a set of interconnected nodes arranged into multiple parallel layers; i.e., an input layer, an arbitrary number of hidden layers and an output layer. Except for the input layer nodes, each node is called a neuron that propagates an output to the next layer through a nonlinear activation function applied over the weighted sum of the received inputs plus a bias factor. Mathematically, this can be formulated as

$$y(t) = f\left(\sum_{x \in \mathcal{F}} \omega_{iu}^{x} \Delta L_{iu}^{x}(t) + T_x\right), \quad (16)$$

where the input element $\Delta L_{iu}^{x}(t)$ is the x path losses statistical metric reported by the i^{th} BS, and $y(t)$ denotes the output

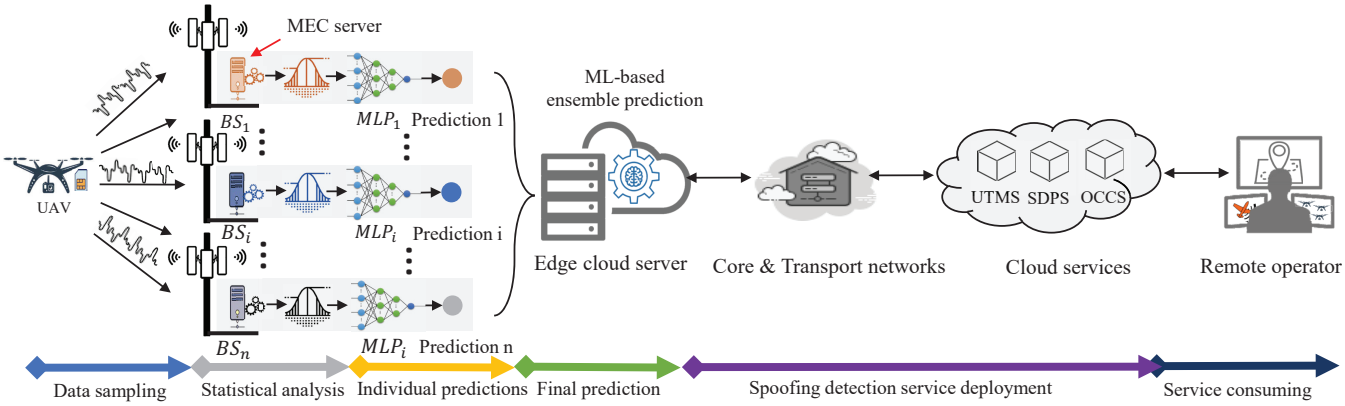


Fig. 4. The MEC architecture for deep ensemble learning-based GPS spoofing detection.

prediction result at time t. Here, $y(t) \in [0, 1]$ denotes the probability whether the position is spoofed or not, where a value close to 1 means that the GPS position is likely to be spoofed. $f(\cdot)$ is the nonlinear activation function, where the Rectified Linear Unit (ReLU) is set as a default activation function. ω_{iu}^{x} stands for the neuron that stores the weight relationship among different statistical metrics and T_x is the threshold $T_u(t)$ of the hypothesis testing. The predication result of each BS will be the input of the ML-based ensemble method described in the next section.

B. Structure of Multi-MLP Ensemble Approach

We propose a MEC architecture that uses deep ensemble learning to aggregate the individual prediction results of the different BSs for GPS spoofing detection. As shown in Fig. 4, this approach consists of four main processes, namely data sampling, statistical analysis, individual prediction, final prediction, in addition two extra processes for deploying and consuming the spoofing detection service.

In the sampling phase, the actual path losses data and the corresponding theoretical path losses are grouped into different groups of E samples. Note that the theoretical path losses are computed based on the reported UAV's position $(\text{Lat}_u, \text{Lon}_u, \text{Alt}_u)$ and the BS's position $(\text{Lat}_i, \text{Lon}_i, \text{Alt}_i)$. A statistical analysis is then performed using MVSK, BOX and WD methods to extract the statistical characteristics of actual and theoretical path losses data. Using the differences between the actual and theoretical statistical metrics as inputs, the trained MLP models will separately predict whether the GPS position is spoofed or not. Finally, the MLPs' prediction results are integrated to provide the final decision on the presence or absence of a spoofing attack based on a preset threshold (See Section V-D1) or ML methods (See Section V-D2). Specifically, the MEC servers are in charge of path loss data collection, statistical analysis, and MLP predictions, while the ensemble learning is conducted on the edge cloud server.

In practice, the spoofing detection service deployment and consumption work in the background to support high prediction performance for the spoofing detection. In fact, the UTM service (UTMS) can deploy spoofing detection services to an edge cloud server through the core and transport networks, and also inform remote operators when the GPS is spoofed through the Operator Command and Control Service (OCCS). The Supplementary Data Provider Service (SDPS) provides meteorological data and other information to the remote operator and UTMS for UAV flight planning and management [4].

In addition to the MLP-based ensemble approach, the Long Short-Term Memory (LSTM) neural network is alternatively used in ensemble architecture. The LSTM-based ensemble architecture allows to extract of the temporal features from path losses and can further improve the detection performance. Compared with MLP-based ensemble architecture, the LSTM-based ensemble approach needs more historical path losses data to train a new model with a spoofing pattern. Moreover, the inherent mobility of UAVs implants high dynamics in the path losses data, which causes too much noise in the long-time path losses spoofing feature and could decrease the detection performance. Although the use of statistical methods can remove the environmental impacts, the LSTM-based ensemble approach needs to add an independent LSTM model after each statistical feature, which increases the size of the LSTM-based ensemble architecture and the storage usage in the MEC server. Regarding the time-consuming and storage usage, the MLP-based ensemble approach is more efficient than the LSTM-based approach because MLP uses a small size of neural network to learn the recurrent spoofing behaviors from the path losses.

C. Statistical-MLP Approach

To eliminate PL data fluctuations caused by environmental changes, we propose a statistical-MLP approach for GPS spoofing detection at BS level. In this approach, a MLP model is built at each BS by training the MLP neural network on dataset comprising the differences of statistical metrics collected locally. The training is carried out using the stochastic gradient descent method for optimizing the MLP neural network parameters (i.e., weights and bias) to minimize the loss function. In this study, we adopt the Adaptive Moment

Estimation (Adam) optimizer [49]. Moreover, the binary cross-entropy loss function is used to assess the performance of the MLP model, which is expressed as

$$B = -\sum_{i=1}^{n} \hat{y}_i \log y_i + (1 - \hat{y}_i) \log(1 - y_i), \tag{17}$$

where B is the binary cross-entropy loss, $\hat{y}_i$ is the i^{th} predicted observation value, and y_i is the i^{th} real observation value. Note that a smaller B corresponds to a better MLP model.

Algorithm 1 summarizes the full process of the proposed statistical-MLP approach, including data sampling, data processing, data labeling, and MLP training and selection. We use the symbol $\|\cdot\|$ to denote the length of a set, the symbol $|\cdot|$ to represent the absolute value of a number, and the symbol $\lceil . \rceil$ to denote the rounding up of a number to its nearest integer. The data sampling procedure produces the grouped data. Initially, the waypoints reported by an UAV and the position of the BS (it connects to) are used to calculate the theoretical PLs (line 2). Thereafter, the PL differences $\Delta\mathcal{L}$ between the actual PLs reported by the BS and the theoretical PLs are computed (line 3). We assume that the planned waypoints $\mathcal{P}$, the reported waypoints $\mathcal{R}$ and the reported PLs are synchronous and have the same length. The PL differences are evenly divided into equal-sized groups, each of which consists of E values of PL differences (line 4). To ensure that the size of the last group is E, the indices of its elements are from $(\|\Delta L\| - E)$ to $\|\Delta L\|$. Besides, the distance between planned waypoints and reported waypoints are calculated and grouped in order to further decide whether the reported GPS is spoofed or not (lines 5–6). The data processing procedure applies the statistical methods defined in sub-section IV-A on each PL differences group to alleviate the environmental impacts and produce the data points for training and testing the MLP models (lines 10–18). The data labeling procedure aims to assign a label y $(1/0)$ to each data point f according to the relationship between the tolerated GPS error dE and the distance between the reported and planned positions (line 21–31). In fact, a data point is labeled as GPS spoofing (i.e., $y = 1$), if the average distance of the corresponding group is greater than the tolerated GPS error. Otherwise, it is labeled as legitimate GPS error (i.e., $y = 0$). It is well known that the performance of a deep neural network model influenced by the tuning of its hyperparameters (e.g., number of layers, the number of neurons in each layer, the batch size, and the learning rate) [50]). To find the most accurate MLP model, the training and selection procedure (lines 34–46) applies a grid search method [51] to determine the optimal hyperparameters' values that lead to the smallest loss error. The search space for finding the optimal hyperparameters for the MLP model is defined in Table IV. For each combination of hyperparameters, the corresponding MLP model is built on the training data set and its performance is evaluated on the unseen testing dataset. The MLP model with the smallest loss error B is used by the BS for detecting GPS spoofing.

An important consideration for assessing the practicality of

Algorithm 1 Statistical-MLP Algorithm.

Input:
$\mathcal{P}$: Planned way points, $\mathcal{P} = \{P_1, \cdots, P_j, P_{j+1}, \cdots\}$.
$\mathcal{R}$: Reported way points, $\mathcal{R} = \{R_1, \cdots, R_j, R_{j+1}, \cdots\}$.
L: BS reported path losses, $L = \{L_1, \cdots, L_j, L_{j+1}, \cdots\}$.
dE: System GPS error tolerance.
$\mathcal{F}$: Statistical methods, $\mathcal{F}$ = {M,V,S,K,Q0,Q1,Q2,Q3,Q4,W}.
E: The number of data points in each group.
γ: Fraction to create bootstrapped training data.
$\mathcal{A}$: hyper-parameters search space.

Output:
$\boldsymbol{c}$: The final MLP model.

1: **procedure** PATHLOSSDATASAMPLING($\mathcal{P}$, $\mathcal{R}$, L)
2: Compute the theoretical path losses $\overline{L}$ using the BS's position, the reported way points $\mathcal{R}$ and Eq. (1);
3: $\Delta L \leftarrow |L - \overline{L}|$; ▷ Differences between the reported and theoretical path losses
4: $\mathcal{L} = \{\mathcal{L}_i\}_{i=1}^{\|\mathcal{L}\|} \leftarrow Divide(\Delta L, \|\mathcal{L}\|)$, where $\|\mathcal{L}\| = \left\lceil \frac{\|\Delta L\|}{E} \right\rceil$
5: $\Delta D = |\mathcal{P} - \mathcal{R}|$; ▷ Distance between $\mathcal{P}$ and $\mathcal{R}$
6: $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^{\|\mathcal{D}\|} \leftarrow Divide(\Delta D, \|\mathcal{D}\|)$, where $\|\mathcal{D}\| = \|\mathcal{L}\|$
7: **return** $\mathcal{L}$, $\mathcal{D}$
8: **end procedure**
9: **procedure** STATISTICALANALYSIS($\mathcal{L}$, $\mathcal{F}$)
10: $\mathbf{X}$: Data points, $\mathbf{X} \leftarrow \{\}$;
11: **for each** $\mathcal{L}_i$ in $\mathcal{L}$ **do**
12: $f \leftarrow \{\}$; ▷ vector of statistical features
13: **for each** $\mathcal{F}_i$ in $\mathcal{F}$ **do**
14: $f \leftarrow f \cup ComputeStaticMetric(\mathcal{L}_i, \mathcal{F}_i)$;
15: **end for**
16: $\mathbf{X} \leftarrow \mathbf{X} \cup \{f\}$;
17: **end for**
18: **return** $\mathbf{X}$
19: **end procedure**
20: **procedure** GROUPDATALABELING($\mathcal{D}$, dE)
21: $\mathbf{Y}$: Data labels, $\mathbf{Y} \leftarrow \{\}$;
22: **for each** $\mathcal{D}_i$ in $\mathcal{D}$ **do**
23: $d = \frac{1}{\|\mathcal{D}_i\|} \sum_{d_j \in \mathcal{D}_i} d_j$;
24: **if** $d > dE$ **then**
25: $y = 1$;
26: **else**
27: $y = 0$;
28: **end if**
29: $\mathbf{Y} \leftarrow \mathbf{Y} \cup \{y\}$;
30: **end for**
31: **return** $\mathbf{Y}$
32: **end procedure**
33: **procedure** MLPTRAINING(($\mathbf{X}, \mathbf{Y}$), γ)
34: Divide $(\mathbf{X}, \mathbf{Y})$ into training data set $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}})$ and testing data set $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ according to γ;
35: $\mathbb{C}$: Classifiers set, $\mathbb{C} \leftarrow \{\}$;
36: **for each** combination $a_i \in \mathcal{A}$ **do**
37: Create the MLP classifier $\mathbb{C}_i$ using the training data set $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}})$;
38: $\mathbb{C} \leftarrow \mathbb{C} \cup \{\mathbb{C}_i\}$;
39: **end for**
▷ MLP model selection
40: $\mathbf{B}$: The set of the loss values, $\mathbf{B} \leftarrow \{\}$;
41: **for each** $\mathbb{C}_i$ in $\mathbb{C}$ **do**
42: Use $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ to evaluate $\mathbb{C}_i$ and obtain the average loss B_i;
43: $\mathbf{B} \leftarrow \mathbf{B} \cup \{B_i\}$;
44: **end for**
45: Get the min loss index k, $k = \text{index}(\min(\mathbf{B}))$;
46: **return** $\boldsymbol{c} \leftarrow \mathbb{C}_k$.
47: **end procedure**

the proposed approach is its time complexity. To this end, we theoretically analyze the time complexity for building the statistical-MLP model. Theorem 1 provides the upper bound of the statistical-MLP algorithm's asymptotic time complexity.

Theorem 1. *The time complexity of building the statistical-MLP algorithm is* $O(5 \times \|\mathcal{P}\| + \|\mathcal{L}\| \times (\|\mathcal{F}\| + 1) + (\gamma \times (\bigtriangledown_{MLP} - 1) + 1) \times \|\mathcal{L}\| \times \|\mathbb{C}\| + 1)$*, where* $\|\mathcal{P}\|$ *is the number of way points in* $\mathcal{P}$*,* $\|\mathcal{L}\|$ *is the number of PL data groups,* $\|\mathcal{F}\|$ *is the number of statistical methods,* γ *is the fraction to create bootstrapped training data,* $\bigtriangledown_{MLP}$ *denotes the time complexity of training each MLP and* $\|\mathbb{C}\|$ *is the number of the MLPs.*

Proof. The time complexity of the statistical-MLP algorithm is the sum of the computational complexities for performing path loss data sampling, statistical analysis and labeling, as well as MLP model training and selection. Thus, we need to calculate the time complexity of each phase.

The time complexity of path loss data sampling is $O(5 \times \|\mathcal{P}\|)$, which includes the time needed for computing the theoretical PLs, calculating and grouping the differences between the theoretical and actual PLs, and finally calculating and grouping the distances between the planned and reported way points. The time complexity of statistical analysis and labeling is $O(\|\mathcal{L}\| \times (\|\mathcal{F}\| + 1))$, which equals to the sum of the time complexity of computing the statistical features of each group and that of labeling each group. Let I, q, and h denote, respectively, the number of training iterations, the number of hidden layers, and the number of neurons in each hidden layer. The time complexity of MLP model training and selection is $O((\gamma \times (\bigtriangledown_{\text{MLP}} - 1) + 1) \times \|\mathcal{L}\| \times \|\mathbb{C}\| + 1)$, where $\bigtriangledown_{\text{MLP}} \approx O(I \times \|\mathcal{F}\| \times q^h)$ according to [52]. Here, the time complexity of the training process is $O(\|\mathcal{L}\| \times \gamma \times \bigtriangledown_{\text{MLP}} \times \|\mathbb{C}\|)$, and that of the selection process is $O(\|\mathcal{L}\| \times (1 - \gamma) \times \|\mathbb{C}\| + 1)$.

Based on the time complexity of these three phases, we obtain that the time complexity of the statistical-MLP is $O(5 \times \|\mathcal{P}\| + \|\mathcal{L}\| \times (\|\mathcal{F}\| + 1) + (\gamma \times (\bigtriangledown_{\text{MLP}} - 1) + 1) \times \|\mathcal{L}\| \times \|\mathbb{C}\| + 1)$. □

D. Multi-MLP Ensemble Approach

To provide the final decision on the presence or absence of GPS spoofing with high accuracy, we propose a multi-MLP ensemble approach for integrating the individual prediction results issued by the MLP models deployed at the BSs. To this end, we consider two strategies for aggregating the individual predictions, namely: (i) A threshold based weighted aggregation, where the final prediction is produced by comparing the weighted combination of individual predictions to a preset threshold; and (ii) a ML-based aggregation, where the final decision is taken by combining the individual predictions through a meta-learner built using different ML techniques.

*1) **Threshold-based Multi-MLP Ensemble Approach**:* This approach aims at finding the best threshold for solving the optimization problem defined in Eq. (15) based on a weighted aggregation of individual predictions. As described in Algorithm 2, the threshold-based multi-MLP ensemble approach comprises three sequential procedures; i.e., the data weighting procedure, the threshold evaluation procedure, and the threshold selection procedure. The data weighting procedure (lines 4–14) aims to assign a weight to the prediction of each ensemble individual MLP model. The weight is proportional to the prediction confidence score of GPS spoofing; that is, the highest weight is assigned to the individual prediction with the highest confidence score. The weights are normalized so that their sum is equal to 1. The ensemble weighted prediction D_i is defined as the sum of the weighted predictions of the individual MLP models (line 10). The threshold evaluation procedure (lines 17–40) uses a set of thresholds and the performance metrics (i.e., missed detection and false alarms)

Algorithm 2 Threshold-based multi-MLP ensemble algorithm.

Input:
- $\mathfrak{D}$: Data set. Each data point in $\mathfrak{D}$ is a vector with size $(n+1)$; that includes n BSs' MLP predictions and one label which is set to 1 if spoofing and 0 if normal.
- Θ: The thresholds set, $\Theta = \{0.00, 0.01, 0.02, ..., 0.99\}$;
- α: The importance coefficient for missed detection.

Output:
- $\mathcal{T}$: The final ensemble threshold for the given date set $\mathfrak{D}$.

1: $\mathbf{X}$: Data points, $\mathbf{X} = \mathfrak{D}[1:n]$;
2: $\mathbf{Y}$: Data labels, $\mathbf{Y} = \mathfrak{D}[n:n+1]$;
3: **procedure** DATAWEIGHTING($\mathbf{X}$)
4: $\quad$ $\mathbf{D}$: The set of weighted MLP predictions, $\mathbf{D} \leftarrow \{\}$;
5: $\quad$ **for each** $X_i \in \mathbf{X}$ **do**
6: $\quad\quad$ $\mathcal{W}$: The weights for X_i, $\mathcal{W} \leftarrow \{\}$;
7: $\quad\quad$ **for each** $x_j \in X_i$ **do**
8: $\quad\quad\quad$ $w_j = x_j / SUM(X_i)$;
9: $\quad\quad\quad$ $\mathcal{W} \leftarrow \mathcal{W} \cup \{w_j\}$;
10: $\quad\quad$ **end for**
11: $\quad\quad$ $\mathbf{D}_i = \mathcal{W} \bigodot X_i$;
12: $\quad\quad$ $\mathbf{D} \leftarrow \mathbf{D} \cup \{\mathbf{D}_i\}$;
13: $\quad$ **end for**
14: $\quad$ **return** $\mathbf{D}$
15: **end procedure**
16: **procedure** THRESHOLDEVALUATION($\mathbf{D}$, $\mathbf{Y}$, Θ)
17: $\quad$ $\mathfrak{R}$: The thresholds test results, $\mathfrak{R} \leftarrow \{\}$;
18: $\quad$ **for each** θ in Θ **do**
19: $\quad\quad$ FP: False positives, $FP \leftarrow 0$;
20: $\quad\quad$ FN: False negatives, $FN \leftarrow 0$;
21: $\quad\quad$ $\widehat{\mathbf{Y}}$: The test results, $\widehat{\mathbf{Y}} \leftarrow \{\}$;
22: $\quad\quad$ **for each** $\mathbf{D}_i$ in $\mathbf{D}$ **do**
23: $\quad\quad\quad$ **if** $\mathbf{D}_i > \theta$ **then**
24: $\quad\quad\quad\quad$ $\hat{y} \leftarrow 1$;
25: $\quad\quad\quad$ **else**
26: $\quad\quad\quad\quad$ $\hat{y} \leftarrow 0$;
27: $\quad\quad\quad$ **end if**
28: $\quad\quad\quad$ $\widehat{\mathbf{Y}} \leftarrow \widehat{\mathbf{Y}} \cup \{\hat{y}\}$;
29: $\quad\quad$ **end for**
30: $\quad\quad$ **for each** ($\hat{y}_i$ in $\widehat{\mathbf{Y}}$) and (y_i in $\mathbf{Y}$) **do**
31: $\quad\quad\quad$ **if** ($\hat{y}_i \neq y_i$) and ($y_i = 1$) **then**
32: $\quad\quad\quad\quad$ $FN \leftarrow FN + 1$;
33: $\quad\quad\quad$ **else if** ($\hat{y}_i \neq y_i$) and ($y_i = 0$) **then**
34: $\quad\quad\quad\quad$ $FP \leftarrow FP + 1$;
35: $\quad\quad\quad$ **end if**
36: $\quad\quad$ **end for**
37: $\quad\quad$ $R_\theta = \alpha \times FN + (1 - \alpha) \times FP$;
38: $\quad\quad$ $\mathfrak{R} \leftarrow \mathfrak{R} \cup \{R_\theta\}$;
39: $\quad$ **end for**
40: $\quad$ **return** $\mathfrak{R}$
41: **end procedure**
42: **procedure** THRESHOLDSELECTION($\mathfrak{R}$, Θ)
43: $\quad$ Get the min value index k, $k = \text{index}(\min(\mathfrak{R}))$;
44: $\quad$ **return** $\mathcal{T} \leftarrow \theta_k$.
45: **end procedure**

defined in (15) to evaluate the ensemble weighted predictions. For each threshold value θ, the procedure first tests whether or not a GPS position is spoofed through the hypothesis testing in Eq. (14), by comparing the ensemble weighted predictions to the preset threshold. If the ensemble weighted prediction is above the threshold, the spoofing hypothesis (i.e., null hypothesis H_0) is accepted and the label 1 is assigned to the test data. Otherwise, the spoofing hypothesis is rejected and the label 0 is associated to the test data. The testing labels $\widehat{\mathbf{Y}}$ are then compared to the actual data labels $\mathbf{Y}$ to assess the detection performance of the multi-MLP ensemble model under the preset threshold. The detection performance is evaluated with respect to the number of false negatives (FNs) and false positives (FPs). FNs refer to missed detections (i.e., $\hat{y}_i = 0$) when the actual GPS positions are spoofed (i.e., $y_i = 1$), while FPs represent the false alarms (i.e., $\hat{y}_i = 1$) when the actual GPS positions are legitimate (i.e., $y_i = 0$). The threshold selection procedure (lines 43–44) uses the detection performance metrics to determine the best threshold. According to Eq. (15), the best threshold is the one minimizing both FNs and FPs. In the following Theorem 2, we give the time complexity of the threshold-based multi-MLP ensemble algorithm.

Theorem 2. *The time complexity of building the Threshold-based multi-MLP ensemble algorithm is* $O((n + \|\Theta\| \times 2) \times \|\mathfrak{D}\| + 1)$. *Here,* n *denotes the number of BSs,* $\|\mathfrak{D}\|$ *is the number of prediction results in* $\mathfrak{D}$ *obtained by Algorithm 1, and* $\|\Theta\|$ *represents the length of the preset thresholds set.*

Proof. The time complexity of the threshold-based multi-MLP ensemble algorithm corresponds to the sum of the time complexity of computing the ensemble weighted predictions and that of thresholds evaluation and selection. The time complexity of calculating the ensemble weighted predictions can be determined as $O(n \times \|\mathbf{X}\|)$. The time complexity of the thresholds evaluation includes the time required for hypothesis testing $\|\Theta\| \times \|\mathfrak{D}\|$ and the time needed for assessing the detection performance $\|\Theta\| \times \|\mathbf{Y}\|$. The threshold selection incurs $O(1)$ time complexity. As $\|\mathbf{X}\| = \|\mathbf{Y}\| = \|\mathfrak{D}\|$, we obtain that the time complexity of building the threshold-based multi-MLP ensemble algorithm is $O((n + \|\Theta\| \times 2) \times \|\mathfrak{D}\| + 1)$. □

*2) **ML-based Multi-MLP Ensemble Approach**:* Finding the best decision threshold to achieve the desired GPS spoofing detection performance is a challenging task, which depends on various factors, including the number of BSs, the training data set and the FNs-FPs trade-off. Thus, an intelligent solution to automatically learn the decision threshold is vital. To this end, we propose a ML-based multi-MLP ensemble approach for integrating the individual prediction results of the MLP models using six different ML methods, including MLP, Decision Trees (DT), Logistic Regression (LR), Support Vector Machine Kernel (SVMK), Support Vector Machine Gamma (SVMG), and Gaussian Naive Bayes (GNB) [52]–[56]. Specifically, the individual prediction results from the outputs of multi-MLPs are the inputs of these ML methods and the trained model with the smallest binary cross entropy loss B is adopted as the predicting model to make a decision on whether the GPS provided by the UAV is spoofed or not.

The proposed ML-based multi-MLP ensemble approach is summarized in Algorithm 3. In the following Theorem 3, we give the time complexity of the ensemble algorithm.

Algorithm 3 ML-based multi-MLP ensemble algorithm.

Input:
- $\mathfrak{D}$: Data set. Each data point in $\mathfrak{D}$ is a vector with size $(n+1)$; that includes n BSs' MLP predictions and one label which is set to 1 if spoofing and 0 if normal.
- $\mathfrak{M}$: Ensemble ML methods, $\mathfrak{M}$ = {MLP, DT,LR, SVMK, SVMG, GNB}.
- γ: Fraction to create bootstrapped training data

Output:
- $\mathfrak{C}$: The final ensemble ML model.

1: $\mathbf{X}$: Data points, $\mathbf{X} = \mathfrak{D}[1:n]$
2: $\mathbf{Y}$: Data labels, $\mathbf{Y} = \mathfrak{D}[n:n+1]$
3: Divide $(\mathbf{X}, \mathbf{Y})$ into training data set $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}})$ and testing data set $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ according to γ
4: **procedure** MODELTRAINING($(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}})$, $\mathfrak{M}$)
5: $\mathcal{M}$: The set of ML meta-models, $\mathcal{M} \leftarrow \{\}$
6: **for each** $\mathfrak{M}_i \in \mathfrak{M}$ **do**
7: Utilize the $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}})$ to train $\mathfrak{M}_i$;
8: $\mathcal{M} \leftarrow \mathcal{M} \cup \{\mathfrak{M}_i\}$;
9: **end for**
10: **return** $\mathcal{M}$
11: **end procedure**
12: **procedure** MODELEVALUATION($(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$, $\mathcal{M}$)
13: $\mathbf{B}$: The set of the Loss values, $\mathbf{B} \leftarrow \{\}$
14: **for each** $\mathcal{M}_i$ in $\mathcal{M}$ **do**
15: Use $(\widehat{\mathbf{X}}, \widehat{\mathbf{Y}})$ to evaluate $\mathcal{M}_i$ and obtain the average loss B_i;
16: $\mathbf{B} \leftarrow \mathbf{B} \cup \{B_i\}$;
17: **end for**
18: **return** $\mathbf{B}$
19: **end procedure**
20: **procedure** MODELSELECTION($\mathbf{B}$, $\mathcal{M}$)
21: Get the min loss index k, $k = \text{index}(\min(\mathbf{B}))$;
22: **return** $\mathfrak{C} \leftarrow \mathcal{M}_k$
23: **end procedure**

Theorem 3. *The time complexity of building the ML-based multi-MLP ensemble algorithm is* $O(\gamma \times \|\mathfrak{D}\| \times \sum_{i=1}^{\|\mathfrak{M}\|} \bigtriangledown_{\mathfrak{M}_i} + (1-\gamma) \times \|\mathfrak{D}\| \times \|\mathfrak{M}\| + 1)$. *Here,* γ *is the fraction to create bootstrapped training data from the MLPs' prediction results* $\mathfrak{D}$ *obtained by Algorithm 1,* $\|\mathfrak{D}\|$ *is the number of prediction results in* $\mathfrak{D}$*, and* $\bigtriangledown_{\mathfrak{M}_i}$ *denotes the time complexity of* i^{th} *ML method* $\mathfrak{M}_i$ *where* $\mathfrak{M}_i \in$ {*MLP, DT, LR, SVMK, SVMG, GNB*}.

Proof. The time complexity of the ML-based multi-MLP ensemble algorithm corresponds to the sum of the time complexity of ML model training and that of ML model evaluation. The time complexity of ML model training can be determined as $O(\gamma \times \|\mathfrak{D}\| \times \sum_{i=1}^{\|\mathfrak{M}\|} \bigtriangledown_{\mathfrak{M}_i})$, where $\gamma \times \|\mathfrak{D}\|$ is the number of the elements of the training data set, $\bigtriangledown_{\text{MLP}} \approx O(I \times n \times q^h)$, n is the number of features, $\bigtriangledown_{\text{DT}} \approx O(n \times \rho)$, ρ is the depth of the tree, $\bigtriangledown_{\text{LR}} \approx O(2 \times n)$ for the classification of two labels, $\bigtriangledown_{\text{SVMK}} \approx O(n)$, $\bigtriangledown_{\text{SVMG}} \approx O(n)$, and $\bigtriangledown_{\text{GNB}} \approx O(2 \times n)$ [52]–[56]. The time complexity of ML model evaluation is $O(\|\mathfrak{D}\| \times \|\mathfrak{M}\|)$. Besides, the time complexity of preparing data and selecting ML model in the training and evaluation processes is $O(1)$. Thus, we obtain that the time

TABLE IV
HYPERPARAMETERS SETTINGS OF MLP MODELS.

Parameter	Value(s)
Number of layers	$1, 2, 3, 4, 5$
Neurons per layer	$10, 20$
Activation function	ReLU
Solver	Adam
Learning rate	$0.001, 0.005, 0.0001, 0.0005$
Batch size	$50, 100, 150$

complexity of ML-based multi-MLP ensemble algorithm is $O(\gamma \times \|\mathfrak{D}\| \times \sum_{i=1}^{\|\mathfrak{M}\|} \bigtriangledown_{\mathfrak{M}_i} + (1-\gamma) \times \|\mathfrak{D}\| \times \|\mathfrak{M}\| + 1)$. □

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performances of the statistical-MLP, threshold-based multi-MLP and ML-based multi-MLP ensemble algorithms, respectively. We also conduct a performance comparison between the threshold and ML-based multi-MLP ensemble algorithms.

A. Simulation Settings

We develop a simulator using Python 3.6 and Tensorflow 2.1 to evaluate the performances of the proposed algorithms. In this simulator, Python is used to set up the simulation platform and Tensorflow is applied to build the ML models. We consider that the nine BSs are evenly distributed in an area of 300×300 m^2, where the distance between any two adjacent BSs is 150 m. The height of the BSs is set to 35 m. The flight height of both the target UAV u and the aerial GPS spoofer is set to 150 m. Initially, u is at the location above the center of the area, and there are a total of 12 potential trajectories (i.e., flight paths) toward different directions. A polling strategy is adopted to choose two alternative paths from the preset twelve paths, and one of them is used as the planned trajectory while the other one is assumed as real time trajectory that can be spoofed or not. Meanwhile, we apply the path loss models defined in [21] to generate the training and evaluating data from the BSs to u. The channel frequency is set to 2.0 GHz.

In Algorithm 1, the number of data points E is drawn from $\{100, 150, 200,\ 250, 300, 350, 400, 450, 500\}$. dE is preset as $\{10, 15, 20, 25\ , 30, 35, 40, 45, 50\}$ to simulate the GPS error tolerance of different scenarios. γ is 0.7, which means that 70% of the data in the data set is used for training the model, and the other 30% is used for its evaluation. It is well known that the performance of MLP is sensitive to hyperparameter settings [57]. Thus, to find the best configuration of the different MLP models in our study, we carry out hyperparameter tuning by varying the learning rate, the number of hidden layers, the number of neurons per hidden layer and the training batch size. Note that the batch size is the number of data points used for each training iteration. The number of MLP layers is varied from 1 to 5 layers, and the number of neurons of the hidden layers is taken from $\{10, 20\}$. Specifically, all MLP models have one input layer with 10 neurons and one output layer with one neuron. The learning rate is drawn from $\{0.001, 0.005, 0.0001, 0.0005\}$. In our simulation, the MLP model with three hidden layers $(10, 20, 10)$ using the learning rate 0.005 achieved the best accuracy performance on both training and validation datasets. All MLP layers use Rectified Linear Unit (ReLU) as their activation function and use Adam as the solver, where the ReLU is used in the hidden layer to prevent the vanishing gradient problem and accelerate the training processes while the Adam solver can work with sparse gradients and use the average of recent magnitudes of the gradients to mitigate noise impacts. The MLP models are built using back propagation method on a data set containing approximately 1 million samples, and each element of the data set consists of a planned way point, a reported way point and a BS position. We set PL observations as $\|L\| \approx 1 \times 10^6$. $\|\mathcal{P}\| = \|\mathcal{R}\| = \|L\| \approx 1 \times 10^6$. The training is performed at most 200 epochs and an early stopping patience of 10 on loss B is applied to prevent overfitting. The model's loss B defined in (17) is calculated for every epoch on a hold out validation set.

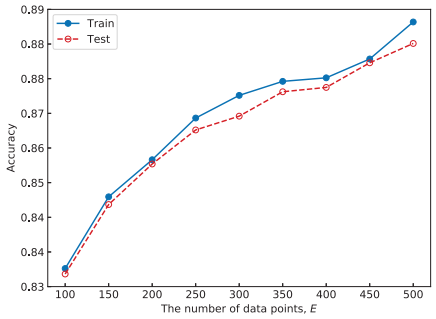
In Algorithm 2, $\mathfrak{D}$ is the data set of different BSs' prediction results, the size of $\mathfrak{D}$ is equal to $(\|\mathcal{P}\| \times n/E)$, and the importance coefficient for missed detection α is set to 0.9. In Algorithm 3, the fraction γ is 0.7 for training data. For the MLP ensemble algorithm, n neurons with one hidden layer is used to integrate each BS decision. DT's depth ρ has the same size as the number of BSs n. According to the logistic regression, the inverse regularization strength of LR is set as 0.01 to reduce overfitting. SVMK model uses the linear kernel with penalty coefficient $\sigma = 0.025$ to increase the fault tolerance of the classifier and avoid overfitting, while SVMG uses radial basis function kernel with parameters $gamma = 2$ and $\sigma = 1$, where $gamma$ decides how much curvature in a decision boundary. Here, a high $gamma$ leads to higher curvature.

B. Performance Metrics

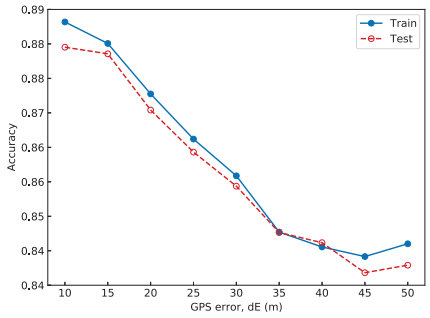
We evaluate the effectiveness of the proposed algorithms according to the commonly used performance metrics, including the total error (TE) of the hypothesis testing (see Eq. (15)), *Accuracy*, *Recall*, and *F1*. The performance metrics are expressed as:

$$\begin{cases} Accuracy = \dfrac{TP + TN}{TP + FP + FN + TN}, \\ Recall = \dfrac{TP}{TP + FN}, \\ Precision = \dfrac{TP}{TP + FP}, \\ F1 = 2 \times \dfrac{Recall \times Precision}{Recall + Precision}, \end{cases} \tag{18}$$

where TP (True Positive) denotes the number of the spoofed positions that are correctly detected, FN (False Negative) represents the number of the spoofed positions that are wrongly detected as normal positions, FP (Flase Positive) is the number of the normal positions that are wrongly detected as spoofed ones, and TN (True Negative) denotes the number of



(a) The effect of E on accuracy under $dE = 15m$.



(b) The effect of dE on accuracy under $E = 150$.

Fig. 5. The accuracy performance of the statistical-MLP algorithm.

the normal positions that are correctly detected. The *F1* metric is used to characterize the trade-off relationship between the precision rate and the recall rate.

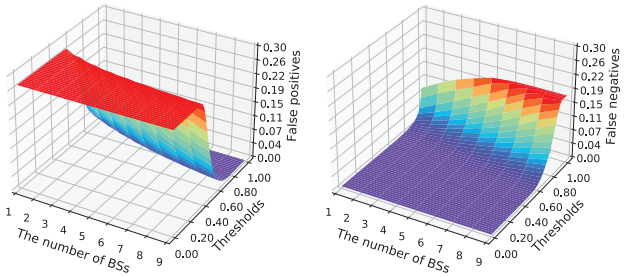
C. Performance Evaluation of Statistical-MLP Approach

We first investigate the impact of the number of data points E on MLP performance under the proposed statistical-MLP approach. The results are summarized in Fig. 5(a) with a setting of GPS error $dE = 15m$. It can be observed from Fig. 5(a) that the MLP accuracy increases as E increases. This is due to the fact that the MLP output performance mainly depends on the input corresponding to these statistical results, where a larger data set results in higher accuracy and precision.

Fig. 5(b) further illustrates how the GPS error dE affects the accuracy under the setting of $E = 150$. We can see from Fig. 5(b) that as GPS error increases, the accuracy decreases. This is because in the environment with a high GPS error, it is more easier for an attacker to counterfeit the GPS position.

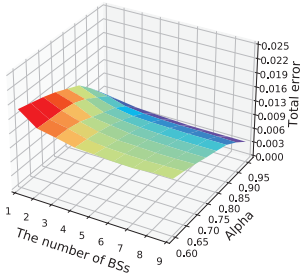
A careful observation from Fig. 5(a) and Fig. 5(b) indicates that the training and testing results have the same increasing/decreasing trend, and the difference between these two results is small. It is notable that a small difference between training and testing results means that the model is well trained without overfitting on testing. We can also observe from Fig. 5 that the accuracy under the proposed statistical-MLP approach is high up to more than 0.83 for all trials. In the next subsection, we will show that this accuracy can be further improved under the ML-based multi-MLP ensemble approach.

D. Performance Evaluation of Multi-MLP Ensemble Approach

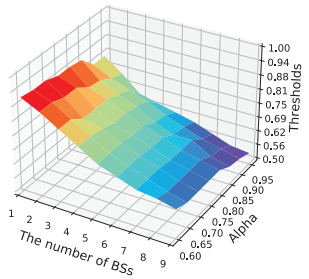


(a) False Positives

(b) False Negatives



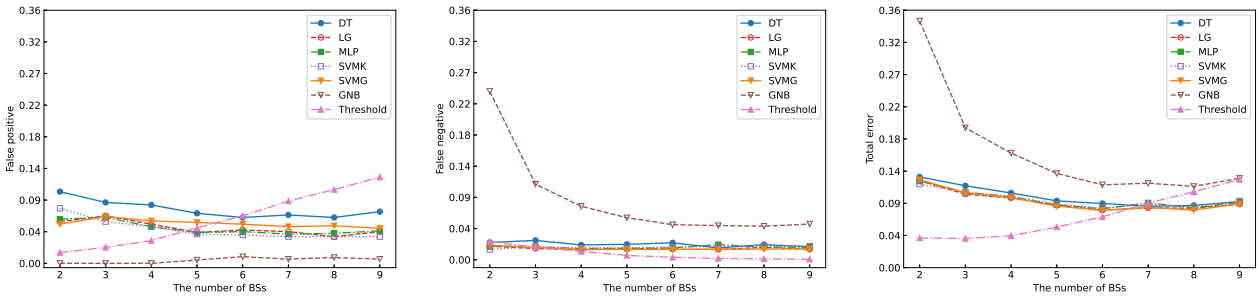
(c) Total Error



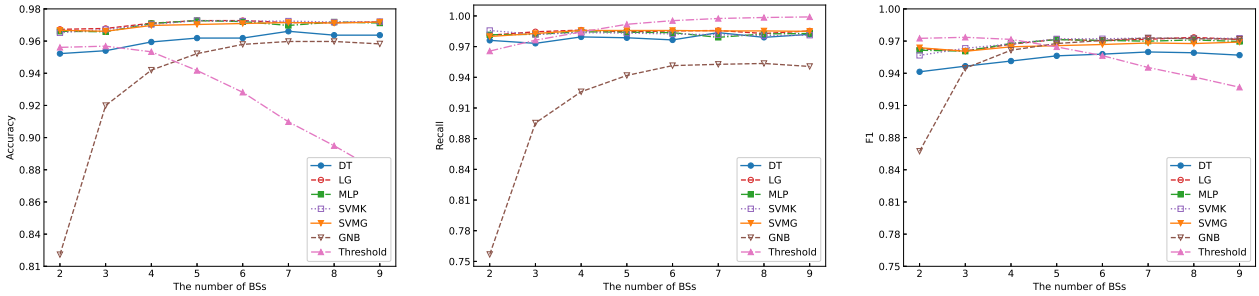
(d) Alpha

Fig. 6. The performances of the threshold-based multi-MLP ensemble algorithm under $dE = 15m$ and $E = 150$.

1) Threshold-based Multi-MLP Ensemble Approach: According to the optimization problem in Eq. (15), the performance of the threshold-based multi-MLP ensemble approach depends on the number of BSs, the threshold value, and the value of α. Fig. 6 illustrates the performance of the threshold-based multi-MLP ensemble approach in terms of *FP*, *FN* and total error. Fig. 6 shows also the influence of the number of BSs and the value of the importance coefficient α assigned to missed detections on the optimal threshold value that yields the smallest total error. The results in Fig.6(a) and Fig.6(b) show that an increase in the threshold value leads to a decrease in FPs and an increase in FNs, respectively. The opposite trends exhibited by FP and FN with respect to the threshold value indicates the interrelation between these two measures. In fact, with lower threshold values, the detection algorithm is likely to detect all spoofed GPS positions, but at the expense of erroneously classifying normal GPS positions as spoofed. On the other hand, higher threshold values make the detection algorithm more permissive, which reduces the number of false alarms but at the price of rising the risk of missing the detection of spoofed GPS positions. Another key observation from Fig.6(a) and Fig.6(b) is that the increase in the number of BSs can lower the FP and FN. This can be explained by the fact that involving more BSs in the detection process will provide more data points which naturally improves the credibility of the ensemble predictions. As introduced in Eq. (15), obtaining the optimal threshold that gives the best detection performance (i.e., the smallest TE) depends on the relative importance of detecting all spoofed positions versus



(a) The effect of the number of BSs on FP (b) The effect of the number of BSs on FN (c) The effect of the number of BSs on TE



(d) The effect of the number of BSs on accuracy (e) The effect of the number of BSs on recall (f) The effect of the number of BSs on on F1

Fig. 7. The performances of multi-MLP ensemble algorithm under $dE = 15m$ and $E = 150$. The thresholds $\theta = [0.76, 0.73, 0.70, 0.66, 0.63, 0.6, 0.58, 0.56]$ with the setting $\alpha = 0.90$, where $\theta[i]$ is the threshold for i BSs scenario.

keeping the number of false alarms low. From results depicted in Fig. 6(c) and Fig. 6(d), it is clear that increasing the importance coefficient of missed detections α leads to lower threshold values to achieve the smallest TEs. Indeed, lower threshold values are required to reduce the FN which allows to minimize their influence on the TE. The results show also the positive impact that the number of BSs have on lowering the TE. As demonstrated in Fig. 6(a) and Fig. 6(b), this positive impact is due to the decrease of FN and FP with the increase in the number of BSs.

2) ML-based Multi-MLP Ensemble Approach: To evaluate the effectiveness of the ML-based multi-MLP ensemble algorithm in detecting the GPS spoofing attack, we measure the fundamental performances in terms of *FP*, *FN*, *TE*, as well as *Accuracy*, *Recall*, and *F1* using the test data set under the setting of $dE = 15m$ and $E = 150$.

We summarize the performance results in Fig. 7. As with the threshold-based multi-MLP ensemble approach, the results depicted in Fig. 7(a) and Fig. 7(b) show that the same observation about the impact of the number of BSs on FP and FN applies; that is, increasing the number of BSs yields to reduced FP and FN. In fact, a higher number of BSs allows to provide more data to the ML meta-model aggregating the individual predictions, which improves the ensemble decision about the spoofing or not of a GPS position. It is worth noting that an exception is noticed for the GNB meta-model, where the increase in the number of BSs slightly increases the generated FP. The reason behind this trend is that GNB is achieving the highest performance in terms of *FP*, with a value less than 0.01, which makes it sensitive to potential errors in predictions provided by some far BSs. Indeed, more the UAV is far away from the BSs, higher the error in the path losses values will be [31].

It can be observed from Fig. 7(d) that the *Accuracy* performance can be significantly improved. For instance, the *Accuracy* is nearly 0.97 using the ensemble model with two BSs, while it is below 0.90 when only using the statistical-MLP algorithm. The reason behind the phenomenon is that the ensemble algorithm takes more evidences from different BSs, which enriched the knowledge for the ensemble algorithm making the final decision. We can also obtain the same conclusion in terms of *Recall* in Fig. 7(e) and *F1* in Fig. 7(f). Furthermore, we observe that the GNB meta-model exhibits the worst performance among the six meta-models. This can be explained by the fact that GNB makes a very strong assumption on the normal distribution of data, which may not necessarily hold for the individual prediction results fed into the GNB meta-model, leading to incorrect detection decisions. For the dataset under study, GNB generates the highest number of missed detections.

We further compare the ML-based algorithms to the threshold-based algorithm as illustrated in Fig.7. It is worth noting that for the threshold-based algorithm, we represent the performance results of the optimal threshold value selected for a given number of BSs when the importance coefficient α is set to 0.9. For instance, the performance results when 3 BSs are considered are the ones obtained with $\theta = 0.73$, while those for 4 BSs are the ones achieved with $\theta = 0.70$.

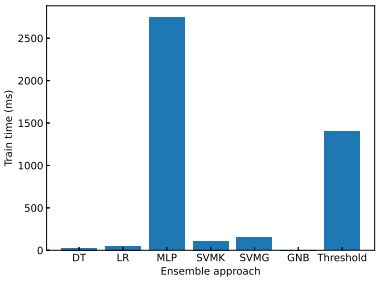
The results in Fig. 7 indicate that the threshold-based algorithm achieves its optimization goal defined in Eq. (15) of minimizing the FN, which also influences positively the *Recall* metric. Although the threshold-based algorithm can reduce the missed detections, it fails in achieving a trade-off among FN, FP, *Accuracy* and *Recall*. Such a balance can be measured using the $F1$ score, which represents the harmonic mean of precision and recall. The *F1*-score values in Fig. 7(f) show that ML-based algorithms outperform the threshold-based algorithm in achieving a better precision-recall balance when the number of BSs is more than 5. From Fig. 7(b), it can be observed that the FN rate for most ML algorithms is less than 2%, which demonstrates the effectiveness of ML-based multi-MLP ensemble approach in protecting the system from GPS spoofing attack while achieving a good balance between precision and recall.

E. Time Complexity Comparison

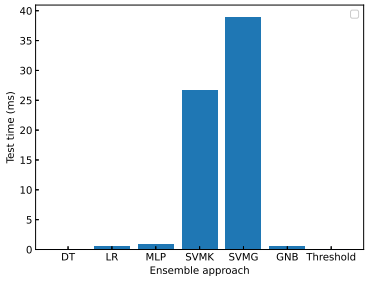
The time for building the detection model (i.e., training time) and the time needed by a trained model to take decision (i.e, testing time) are critical indicators of performance evaluation [50]. Specifically, a longer training time will bring more latency for updating the threshold value or the ML model in a dynamic environment in order to adapt to changes in the path loss model, while a longer testing time introduces more latency for the spoofing detection. In what follows, we discuss and compare the training and testing time complexity of the proposed multi-MLP ensemble approaches. For the threshold-based approach, the training time represents the time required to find the optimal threshold.

From the theoretical analysis conducted in Section V, the time complexity of the threshold-based algorithm is $O((n + \|\Theta\| \times 2) \times \|\mathfrak{D}\|)$, and the complexity of training and testing an ML-based algorithm is of $O(\gamma \times \|\mathfrak{D}\| \times \bigtriangledown_{\mathfrak{M}_i} + (1-\gamma) \times \|\mathfrak{D}\|)$, where $\bigtriangledown_{\mathfrak{M}_i}$ is of $O(n)$ for SVMG and SVMK, $O(2 \times n)$ for LR and GNB, $O(n \times \rho)$ for DT, and $O(I \times n \times q^h)$ for MLP. It can be seen that the time complexity of the threshold-based algorithm is much higher than that of the ML-based algorithms excluding the MLP-based algorithm. This is because the number of thresholds $\|\Theta\|$, equal to 100 in this study, is much bigger than $2 \times n$ and still higher than n^2 for $n \in [1, 9]$. The higher complexity of MLP-based algorithm is due to the fact its training time depends not only on n, but also on the number of hidden layers q, the number of neurons per layer h, and most importantly the number of training epochs I (set to 200 in our simulation).

We also measured the training time and testing time via simulations using a physical machine with 4-cores Intel's Core $1.6Ghz$ CPU and $16GB$ RAM. Fig. 8 illustrates the obtained results. One interesting finding observed from Fig. 8(a) indicates that the training time of the MLP model is significantly higher than that of the other ML-based algorithms. An intuitive explanation is that the MLP model may need more time to update the neurons parameters to provide a better performance. Another important finding from Fig. 8(b) is that the testing time of the SVMK and SVMG is higher than that of the other ML-based algorithms. This is because these SVMK and SVMG models are not incremental; i.e., all the test data are analyzed at once. Regarding the threshold-based algorithm, it spends 1400ms each time on finding the best threshold, which is also much higher than the training time consumed by most of the ML-based algorithms, except the MLP. For the testing time, the threshold-based algorithm only needs to compare the test values with the thresholds, which is of lower complexity compared with all ML-based algorithms.



(a) The training time of ML-based algorithms



(b) The testing time of ML-based algorithms

Fig. 8. Training and testing time consumption of MLP ensemble algorithms.

Based on the above performance evaluations, we find that the ML-based multi-MLP ensemble approach could achieve a significant improvement on the GPS spoofing detection performance compared with the statistical-MLP algorithm and the threshold-based multi-MLP ensemble approach. Meanwhile, a flexible combination of the statistical-MLP algorithm and other ML methods could reduce the spoofing detection latency, and achieve a real time GPS spoofing detection.

VII. Conclusion and Future Work

This paper explored the spoofing detection performance for cellular-connected UAV based on the ML methods. To this end, a constrained optimization problem was proposed to formulate it. To solve the nonlinear and nonconvex optimization problem, we first proposed a statistical-MLP algorithm to independently predict the spoofing probability of GPS position at each BS. For an improvement of detection performance, we further proposed a multi-MLP ensemble algorithm to integrate these independent prediction results using six different ML models, respectively. Remarkably, our proposed algorithm can achieve above 97% accuracy rate under two BSs, while it can

still achieve at least 83% one under only one BS. Furthermore, the testing time under the four types of ML models (i.e., DT, LG, MLP, and GNB) is very short; less than 5 ms for 3×10^4 test data points.

Despite its demonstrated effectiveness in detecting GPS spoofing in an energy-efficient way, the proposed solution is designed for one single cellular-connected UAV and does not consider the case of a UAV swarm. Although our approach can work with multi-UAVs, it requires running multiple spoofing detection models on MEC and edge cloud servers, one per UAV, which leads to less cooperation among the detection procedures. When a large UAV swarm connects with one single BS, it takes too much computation on the BS MEC server and may lead to congestion in the detection system due to the limited computation resources. To support the swarm spoofing detection and further improve the detection performance, the Graphic Neural Network (GNN) will be introduced into the MEC server to detect the GPS spoofing by checking the authenticity of the GPS positions reported by the swarm. Specifically, the GNN is used to compute the similarity between the swarm GPS formation and the swarm communication formation. The swarm GPS location formation is similar to the swarm communication formations without GPS spoofing. Otherwise, the GPS is spoofed.

VIII. Acknowledgement

This work was supported by the European Union's Horizon 2020 Research and Innovation Program through the 5G!Drones Project under Grant 857031, the INSPIRE-5Gplus project under Grant 871808, the Academy of Finland 6Genesis project under Grant 318927, and the National Natural Science Foundation of China under Grants 61972308, 61941114 and 61962033.

References

[1] B. Alzahrani, O. S. Oubbati, A. Barnawi, M. Atiquzzaman, and D. Alghazzawi, "Uav assistance paradigm: state-of-the-art in applications and challenges," *Journal of Network and Computer Applications*, vol. 166, p. 102706, 2020.

[2] U. Market, "Unmanned aerial vehicle (UAV) market share forecast to 2026," https://www.marketsandmarkets.com/Market-Reports/unmanned-aerial-vehicles-uav-market-662.html., January 2022.

[3] J.-P. Yaacoub, H. Noura, O. Salman, and A. Chehab, "Security analysis of drones systems: attacks, limitations, and recommendations," *Internet of Things*, vol. 11, p. 100218, 2020.

[4] FAA, "Unmanned aircraft system traffic management, federal aviation administration (FAA)," https://www.marketsandmarkets.com/Market-Reports/unmanned-aerial-vehicles-uav-market-662.html., January 2021.

[5] B. Ly and R. Ly, "Cybersecurity in unmanned aerial vehicles (uavs)," *Journal of Cyber Security Technology*, vol. 5, no. 2, pp. 120–137, 2021.

[6] A. Shafique, A. Mehmood, and M. Elhadef, "Detecting signal spoofing attack in UAVs using machine learning models," *IEEE Access*, vol. 9, pp. 93 803–93 815, 2021.

[7] X. Wei, M. Aman, and B. Sikdar, "Exploiting correlation among gps signals to detect gps spoofing in power grids," *IEEE Transactions on Industry Applications*, 2021.

[8] H. Sathaye, G. LaMountain, P. Closas, and A. Ranganathan, "Semperfi: anti-spoofing gps receiver for UAVs," in *Network and Distributed Systems Security (NDSS) Symposium 2022*, 2022.

[9] S. Daneshmand, A. Jafarnia-Jahromi, A. Broumandan, and G. Lachapelle, "A low-complexity GPS anti-spoofing method using a multi-antenna array," *a a*, vol. 2, p. 2, 2012.

[10] J. Magiera and R. Katulski, "Detection and mitigation of GPS spoofing based on antenna array processing," *Journal of applied research and technology*, vol. 13, no. 1, pp. 45–57, 2015.

[11] M. L. Psiaki, B. W. O'Hanlon, J. A. Bhatti, D. P. Shepard, and T. E. Humphreys, "GPS spoofing detection via dual-receiver correlation of military signals," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 4, pp. 2250–2267, 2013.

[12] B. W. O'Hanlon, M. L. Psiaki, J. A. Bhatti, D. P. Shepard, and T. E. Humphreys, "Real-rime GPS spoofing detection via correlation of encrypted signals," *Navigation*, vol. 60, no. 4, pp. 267–278, 2013.

[13] Z. Wu, R. Liu, and H. Cao, "Ecdsa-based message authentication scheme for beidou-ii navigation satellite system," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 4, pp. 1666–1682, 2018.

[14] Z. Wu, Y. Zhang, and R. Liu, "Bd-ii nma&ssi: An scheme of anti-spoofing and open beidou ii d2 navigation message authentication," *IEEE Access*, vol. 8, pp. 23 759–23 775, 2020.

[15] K. Zhang, E. G. Larsson, and P. Papadimitratos, "Protecting gnss open service navigation message authentication against distance-decreasing sttacks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 2, pp. 1224–1240, 2021.

[16] G. W. Hein, F. Kneissl, J.-A. Avila-Rodriguez, and S. Wallner, "Authenticating GNSS: proofs against spoofs, part 2," *InsideGNSSS*, pp. 71 – 78, Sept.-Oct. 2007.

[17] D. Schmidt, K. Radke, S. Camtepe, E. Foo, and M. Ren, "A survey and analysis of the GNSS spoofing threat and countermeasures," *ACM Computing Surveys*, vol. 48, no. 4, pp. 64:1 – 64:31, May 2016.

[18] K. Wesson, M. Rothlisberger, and T. Humphreys, "Practical cryptographic civil GPS signal authentication," *NAVIGATION: Journal of the Institute of Navigation*, vol. 59, no. 3, pp. 177–193, 2012.

[19] T. Liu, A. Hojjati, A. Bates, and K. Nahrstedt, "Alidrone: enabling trustworthy proof-of-alibi for commercial drone compliance," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2018, pp. 841–852.

[20] 3GPP TS 23.754, "Study on supporting unmanned aerial systems (UAS) Connectivity, Identification and Tracking," Dec. 2018.

[21] 3GPP TR 36.777, "Enhanced LTE support for aerial vehicles," Dec. 2017.

[22] 3GPP TS 22.825, "Study on remote identification of unmanned aerial Systems (UAS)," Sept. 2018.

[23] 3GPP TS 33.256, "Security aspects of uncrewed aerial systems (UAS)," Mar. 2022.

[24] Y. Dang, C. Benzaïd, Y. Shen, and T. Taleb, "GPS spoofing detector with adaptive trustable residence area for cellular based-UAVs," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[25] Y. Dang, C. Benzaïd, B. Yang, and 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.

[26] I. Fernandez and V. Rijmen and T. Ashur and P. Walker and G. Seco and J. Simon and C. Sarto and D. Burkey and O. Pozzobon , "Galileo navigation message authentication specification for signal-in-space resting - v1.1." European Commission, Jun. 2018.

[27] C. Benzaid, K. Lounis, A. Al-Nemrat, N. Nadjib, and M. Alazab, "Fast authentication in wireless sensor networks," *Future Generation Computer Systems*, vol. 55, pp. 362 – 375, Feb. 2016.

[28] J.-H. Lee, K.-C. Kwon, D.-S. An, and D.-S. Shim, "GPS spoofing detection using accelerometers and performance analysis with probability of detection," *International Journal of Control, Automation and Systems*, vol. 13, no. 4, pp. 951–959, 2015.

[29] K.-C. Kwon and D.-S. Shim, "Performance analysis of direct GPS spoofing detection method with accelerometer," *Sensors*, vol. 20, no. 4, p. 954, 2020.

[30] Z. Feng, N. Guan, M. Lv, W. Liu, Q. Deng, X. Liu, and W. Yi, "An efficient UAV hijacking detection method using onboard inertial measurement unit," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 17, no. 6, pp. 1–19, 2018.

[31] G. Oligeri, S. Sciancalepore, O. A. Ibrahim, and R. Di Pietro, "Drive me not: GPS spoofing detection via cellular network: (architectures, models, and experiments)," in *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, 2019, pp. 12–22.

[32] F. Formaggio, S. Ceccato, F. Basana, N. Laurenti, and S. Tomasin, "Gnss spoofing detection techniques by cellular network cross-check

in smartphones," in *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*, 2019, pp. 3904–3916.
[33] Y. Dang, C. Benzaid, Y. Shen, and T. Taleb, "GPS spoofing detector with adaptive trustable residence area for cellular based-UAVs," in *Proc. of IEEE Globecom*, Dec. 2020.
[34] M. R. Manesh, J. Kenney, W. C. Hu, V. K. Devabhaktuni, and N. Kaabouch, "Detection of GPS spoofing attacks on unmanned aerial systems," in *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–6.
[35] E. Shafiee, M. Mosavi, and M. Moazedi, "Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers," *The Journal of Navigation*, vol. 71, no. 1, pp. 169–188, 2018.
[36] S. Semanjski, A. Muls, I. Semanjski, and W. De Wilde, "Use and validation of supervised machine learning approach for detection of gnss signal spoofing," in *International Conference on Localization and GNSS (ICL-GNSS)*, 2019.
[37] F. Gallardo and A. P. Yuste, "SCER spoofing attacks on the galileo open service and machine learning techniques for end-user protection," *IEEE Access*, vol. 8, pp. 85 515–85 532, 2020.
[38] A. Coster and A. Komjathy, "Space weather and the global positioning system," *Space Weather*, vol. 6, no. 6, pp. 1–6, 2008.
[39] H. Hu and N. Wei, "A study of GPS jamming and anti-jamming," in *2009 2nd international conference on power electronics and intelligent transportation system (PEITS)*, vol. 1. IEEE, 2009, pp. 388–391.
[40] H. Zhu, M. L. Cummings, M. Elfar, Z. Wang, and M. Pajic, "Operator strategy model development in UAV hacking detection," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 6, pp. 540–549, 2019.
[41] A. A. Cabrera-Ponce and J. Martinez-Carranza, "Aerial geo-localisation for mavs using posenet," in *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*. IEEE, 2019, pp. 192–198.
[42] K. Amer, M. Samy, M. Shaker, and M. ElHelw, "Deep convolutional neural network based autonomous drone navigation," in *Thirteenth International Conference on Machine Vision*, vol. 11605. International Society for Optics and Photonics, 2021, p. 1160503.
[43] K. Xiao, J. Zhao, Y. He, C. Li, and W. Cheng, "Abnormal behavior detection scheme of UAV using recurrent neural networks," *IEEE Access*, vol. 7, pp. 110 293–110 305, 2019.
[44] J. Jung and S. Nag, "Automated management of small unmanned aircraft system communications and navigation contingency," in *AIAA Scitech 2020 Forum*, 2020, p. 2195.
[45] O. Bekkouche, M. Bagaa, and T. Taleb, "Toward a utm-based service orchestration for UAVs in mec-nfv environment," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.
[46] D. Joanes and C. Gill, "Comparing measures of sample skewness and kurtosis," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 47, no. 1, pp. 183–189, 1998.
[47] D. F. Williamson, R. A. Parker, and J. S. Kendrick, "The box plot: a simple visual method to interpret data," *Annals of internal medicine*, vol. 110, no. 11, pp. 916–921, 1989.
[48] S. Vallender, "Calculation of the wasserstein distance between probability distributions on the line," *Theory of Probability & Its Applications*, vol. 18, no. 4, pp. 784–786, 1974.
[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.
[50] C. Benzaid and T. Taleb, "AI-driven zero touch network and service management in 5G and beyond: challenges and research directions," *IEEE Network Magazine*, vol. 34, no. 2, pp. 186 – 194, March/April 2020.
[51] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: a big comparison for nas," *arXiv preprint arXiv:1912.06059*, 2019.
[52] M. J. Kearns, *The computational complexity of machine learning*. MIT press, 1990.
[53] H. Buhrman and R. De Wolf, "Complexity measures and decision tree complexity: a survey," *Theoretical Computer Science*, vol. 288, no. 1, pp. 21–43, 2002.
[54] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (svm) in libsvm," *International journal computer and application*, vol. 128, no. 3, pp. 28–34, 2015.
[55] S. Suthaharan, "Support vector machine," in *Machine learning models and algorithms for big data classification*. Springer, 2016, pp. 207–235.
[56] M. Ontivero-Ortega, A. Lage-Castellanos, G. Valente, R. Goebel, and M. Valdes-Sosa, "Fast gaussian naïve bayes for searchlight classification analysis," *Neuroimage*, vol. 163, pp. 471–479, 2017.
[57] M. Claesen and B. De Moor, "Hyperparameter search in machine learning," *arXiv preprint arXiv:1502.02127*, 2015.