IEEE INFOCOM 2024 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) | 979-8-3503-8447-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/INFOCOMWKSHPS61880.2024.10620818

Detecting Stealthy GPS Spoofing Attack Against UAVs Using Onboard Sensors

Anthony Finn†, Mengjie Jia‡, Yanyan Li†, Jiawei Yuan‡
†Department of Computer Science and Information Systems, California State University San Marcos, USA
‡Department of Computer & Information Science, University of Massachusetts Dartmouth, USA
finn023@csusm.edu, mjia@umassd.edu, yali@csusm.edu, jyuan@umassd.edu

***Abstract*—Unmanned aerial vehicles (UAVs), known as drones, have gained significant popularity across various military, civilian, and commercial applications. Given the fact that many UAV operations rely on the Global Positioning System (GPS), they inevitably become susceptible to GPS spoofing attacks. In recent years, AI-enabled detection approaches toward UAV GPS spoofing attacks have increasingly received research attention. Therefore, it is crucial to have a systematical understanding of GPS spoofing attacks and collect a comprehensive and quality data set in the construction of effective AI-enabled detection.**

This paper aims to collect a large dataset of UAV flights under normal and attack scenarios and design an effective detection approach for stealthy UAV GPS spoofing attacks using onboard sensors and machine learning. 30 different features from 4 onboard UAV sensors are extracted in constructing effective AI models. On top of that, we examined different deep learning and machine learning models by fusing important features from our analysis. Our evaluation results in different flight scenarios demonstrated the effectiveness of our proposed approach, in which a high detection accuracy up to 98.7% and a fast detection time of 0.5 second can be achieved using the XGBoost model.

***Index Terms*—UAV security, GPS spoofing, stealthy attack, machine learning, AI-enabled detection, onboard sensors**

I. Introduction

Motivated by the advantages of UAVs in terms of high mobility, ease of deployment, and rich sensing capabilities [1], [2], the adoption of UAVs has become increasingly prevalent in various military missions, civil industries, and personal use, such as real-time monitoring, search-and-rescue operations, wireless coverage, remote sensing, delivery services, security, and surveillance [1]. Although UAV applications provide many benefits, it also faces various security risks, among which GPS spoofing attacks [3] have a significant impact as most UAVs rely on GPS signals for positioning and navigation. For example, an attacker can spoof a GPS signal by transmitting a high power signal using HackRF which overwrites the original one. Through GPS spoofing, an attacker could hijack an UAV and deviate it from its intended flight trajectory.

To detect and mitigate GPS spoofing attacks against UAVs, machine and deep learning based approaches have become a prevalent trend with the rapid development of AI technologies in both algorithms and hardware [4]–[7]. These approaches typically train a model to predict if the received GPS signal is normal or not. Although these approaches have demonstrated their effectiveness in the detection of simple GPS spoofing attacks, e.g., injecting random GPS values, sophisticated attackers can still bypass the detection with strategic stealthy attacks [8], [9]. Specifically, attackers can continuously introduce a small amount of attacking values, which is not sensitive enough to trigger the abnormal alarm but gradually affect the position of UAVs by confusing the UAVs to mistakenly apply their built-in position adjustment [10].

This paper investigates the detection of stealthy UAV GPS spoofing attacks by leveraging onboard UAV sensor data using a sensor-fusion approach. Sensor-fusion is advantageous over a single sensor based approach as it increases the redundancy and robustness of the system but also provides complementary information that helps identify anomalies or discrepancies between sensor readings. This paper analyzes 30 features from 4 onboard sensors that are typically available on most UAVs to obtain a better understanding of their contribution towards the detection of stealthy GPS spoofing attacks. We then evaluate the identified sensor features with different machine learning and deep learning models to detect stealthy UAV GPS spoofing attacks. Our results demonstrated that the integration of sensor-fusion and XGBoost machine learning model achieves the best performance in different flight scenarios. The contributions of this paper are as follows:

- This paper systematically investigates the UAV stealthy GPS spoofing attack in a PX4-based simulation platform and collects a large dataset of UAV flights under both attack and non-attack scenarios from multiple onboard sensors in a time synchronized manner.
- This paper proposes an effective detection system that employs machine learning models, i.e., XGBoost on the 30 features extracted from 4 onboard UAV sensors. The proposed system can achieve a high detection accuracy of 98.7% on common UAV flights and a fast detection time of 0.5 second on low-frequency 5 hz sensor data.
- This paper evaluates 30 features and multiple machine learning and deep learning models on their effectiveness of detecting stealthy GPS spoofing attacks. This paper also evaluates the impact of time windows, flight plans and choice of sensors on the detection accuracy.

The rest of the paper is structured as follows: We review and discuss related work in Section II. In Section III, we present the construction and methodology of the detection system. We evaluate the performance of the detection system in Section IV, and we conclude the paper in Section V.

979-8-3503-8447-5/24/$31.00 ©2024 IEEE

TABLE I
COMPARISON OF EXISTING WORKS USING ONBOARD SENSORS

Paper	*Type*	*# Sample*	*Public*	*Attack*	*Sensor Used*	*# Features*	*ML Models*	*Accuracy*	*Detection Type*
[11]	Simulation	-	N	Simple	IMU, GPS, B, M	8	LSTM	78%	Flight Path Prediction
[12]	Simulation	40,000	N	Simple	GPS only	3	Linear Regression	N/A	Flight Path Prediction
[13]	Physical	-	N	Simple	IMU, GPS	N/A	Math Equation	100%	Spoofing Detection
[4]	Physical	33,000	N	Simple	IMU, GPS	N/A	XGBoost	96.3%	Spoofing Detection
[6]	Physical	5,303	R	Simple	IMU, GPS, B, M	45	RF + XGBoost	99.7%	Spoofing Detection
[14]	Physical	10,296	R	Simple	IMU, GPS, B, M	12	XGBoost	97.7%	Spoofing Detection
[15]	Physical	33,056	N	Simple	GPS only	3	1DCNN	F1: 99%	Spoofing Detection
[16]	Physical	-	N	Advanced	GPS only	6	Statistical Model	90%	Spoofing Detection
[17]	Physical	7,699	N	Simple	IMU, GPS, B, M	36	CNN-LSTM	99.4%	Spoofing Detection
Ours[1]	Simulation	187,388	Y	Advanced	IMU, GPS, B, M	30	XGBoost	98.7%	Spoofing Detection

II. RELATED WORKS

In this section, we will examine previous research studies and their proposed methods for detecting GPS spoofing. With the increasing prevalence of UAVs, there is a need for reliable security in UAV operations. Researchers have employed a variety of strategies to address this attack [4], [6], [11]–[17].

Many typical ML and non-ML based solutions can be categorized into two types of methods, i.e, flight path prediction, and spoofing classification. Flight path prediction attempts to model the drone's in-air trajectory and determine if the drone is flying in the correct direction, or if the drone is in the right position. Spoofing classification attempts to find anomalies and patterns in attacked flight scenarios. In this work, we focus on spoofing detection using UAV onboard sensor data.

Table I offers a brief summary of existing research endeavors, characterizing their proposed methodologies and respective outcomes. Within the "Public" column, designations such as 'Y' signifies availability, 'N' denotes unavailability, and 'R' indicates available upon request for the data. In the "Attack" column, the descriptor 'simple' signifies a limited exploration of GPS spoofing attacks, whereas 'advanced' denotes a more stealthy and sophisticated attempt at GPS spoofing. Additionally, in the "Sensor Used" column, 'B' designates the usage of barometer sensor, and 'M' refers to magnetometer sensor.

Spoofing detection approaches [4], [6], [13]–[17] typically collect data from one or multiple onboard sensors. They analyze the sensor outputs, extract useful features and perform classification based on machine learning models. For instance, Z. Feng et al. [4] utilized IMU, i.e., accelerometer and gyroscope, and GPS signal, to determine if the UAV is under attack. Typically machine learning is adopted to monitor the flight status of UAVs. Many studies have achieved very good results utilizing sensor data-based approaches. Several studies have achieved great success with machine learning models like XGBoost, and LSTM. However, a notable limitation is the absence of a publicly available, comprehensive, and extensive data set. The effectiveness of these approaches relies heavily on the quality and comprehensiveness of the data set. The data set should contain quality normal and attack data. Additionally, the GPS spoofing attack has not been thoroughly investigated. Many studies conduct GPS spoofing attacks by injecting white noise signal data, or random GPS messages to a UAV. Many research works fail to create an attack that can slowly drift a UAV to a different location without being detected by existing countermeasures such as Extended Kalman Filter (EKF).

EKF is a state estimator that analyzes sensor outputs and offers estimates for various state parameters of a UAV, encompassing rotation, velocity, position, sensor biases, and magnetic field components. The UAV's state is forward propagated, generating new predictions based on previous states [18]. If the output of one or multiple sensors deviates far from a predicted value, then EKF will notify UAV about the abnormality.

Flight path prediction [11], [12], while valuable for certain applications, is inherently ill-suited as a mechanism for detecting GPS spoofing due to its limitations in its non-linear forecasting capabilities. It is challenging to accurately anticipate a drone's trajectory solely based on historical data or pre-established patterns. Furthermore, unforeseen environmental factors, such as erratic wind conditions, introduce additional unpredictability, rendering flight path prediction methodologies ineffective in capturing the nuances of a drone's movement under variable circumstances. Consequently, reliance on flight path prediction for GPS spoofing detection proves inadequate, emphasizing the need for more robust and adaptive countermeasures.

III. METHODOLOGY

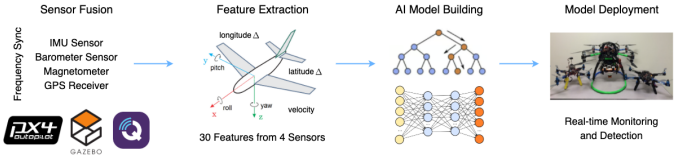

Fig. 1. Overview of Detection Pipeline

In this section, we present the details of our GPS spoofing detection methodology. Fig. 1 illustrates an overview of the detection pipeline, encompassing four main stages: (1) the construction of plugins and GPS spoofing attacks in the PX4 and Gazebo based simulation environment; (2) the collection of data from four sensors in both normal and attack scenarios; (3) the extraction of effective features; and (4) the selection of machine learning models and the detection of new flight data.

[1]https://github.com/anthony-finn/UAV-GPS-Spoofing-Dataset

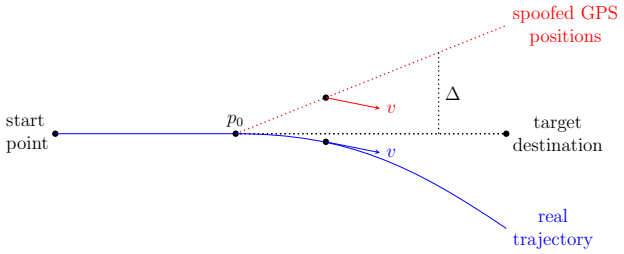

Fig. 2. Stealthy GPS spoofing attack studied in this work

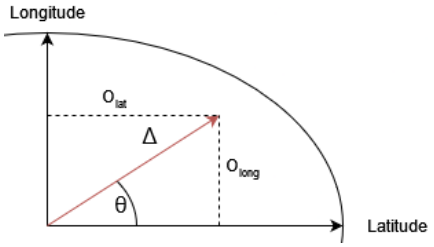

Fig. 3. Latitude and longitude offset calculation visualization

A. GPS Spoofing Attack

The objective of the UAV is to fly from its starting position to a designated target destination. As the UAV reaches position p_0, an attacker launches GPS spoofing on the UAV. The attacker intends to deviate the UAV from its planned flight trajectory while preventing the UAV from detecting the attack. A sophisticated attacker can achieve this by incrementally offsetting the UAV's reported GPS position. The attacker adjusts the offset according to the UAV's sampled velocity. The Δ in the illustration is the maximum latitude and longitude offset in radians which is increased slowly towards the maximum to avoid detection from the detection system. Fig.2 illustrates the simple idea behind our GPS hijacking spoofing attack. The red line shows the spoofed GPS position, while the blue line shows the ground-truth position of the UAV. The UAV's spoofed GPS location is at the target location, while the UAV's actual location is not. Detecting the attack is challenging as the linear movement of the GPS position can be attributed to various external factors, such as wind or GPS inaccuracies, making it difficult to distinguish the attack from other influences.

To launch a sophisticated attack, we compute the 2D latitude longitude GPS heading vector from the position difference over a sample time; this measurement is the sampled velocity. To calculate the heading vector, we extract the north and east components of the sample velocity and normalize it to get the direction vector $\begin{pmatrix} v_e & v_n \end{pmatrix}$. For instance, to initiate a GPS spoofing attack with a displacement of Δo radians from the starting position and a rotation of θ radians counterclockwise from the heading vector, the following formula is used:

$$\Delta o \cdot \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} v_e \\ v_n \end{bmatrix} = \begin{bmatrix} o_{lat} \\ o_{long} \end{bmatrix}$$

The resulting vector is the modified offsets for both latitude (o_{lat}) and longitude (o_{long}) directions. To maintain the spoofed GPS offset position during the attack, we use the sampled heading vector to compensate for any changes in the UAV's direction of movement. Otherwise, the attack could be detected easily. The latitude and longitude directions have distinct scales, and their offsets vary. Although the formula does not account for this scale, it becomes insignificant when the delta offset is minimal. Fig. 3 shows the idea of the offset calculation in both longitude and latitude directions.

When running our attack program `gps_spoofer`, attackers need to provide the `offset` and `degree` variables, where the `offset` is the total amount of latitude or longitude radians to offset the GPS position, and the `degree` is the direction to offset the GPS position in. We reset the `offset` before starting an attack to prevent large jumps in the GPS position information. The earlier matrix multiplication formula provides the o_{lat} and o_{long} variables in the attack program. The attack program also has a small `increment` variable defined where it determines the offset increment each time the GPS signal is received. With a faster GPS update frequency, the offset increment must be set to a smaller value to avoid state estimate based detections such as EKF.

B. Multi-Sensor Data Collection and Labeling

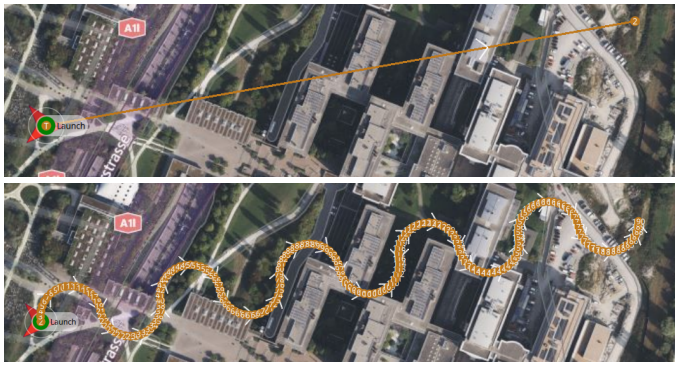

Fig. 4. Example of two generated flight plans

To collect both normal and attacked UAV sensor data, we leverage our UAV cybersecurity simulation platform [19], which is based on QGroundControl [20], PX4-SITL [21], and Gazebo-Classic [22]. We study three flight trajectories: moving in straight, curved, or random paths. To generate data for attacked and non-attacked flights, we developed a program to create flight plans and upload them through QGroundControl utilizing MAVSDK [23] and running them utilizing the PX4-SITL Gazebo-Classic simulation platform. Fig. 4 depicts two sample flight plans generated from the program used in this study. The upper image illustrates a straight flight plan, whereas the lower image illustrates a curved flight trajectory. The random flight trajectory is a combination of both curved and straight flight plans. We developed an extension to the GPS plugin provided by the Gazebo platform that implements

the attack strategy presented in section III-A. We create two attack samples and one non-attack sample for each flight plan. Each attack is launched perpendicular to the UAV's heading angle. Overall we have 360 flights collected that encompass hours of flight time.

When capturing time-series data from UAV sensors, we segment the data into smaller time intervals or windows. There is a correlation between detection time and the length of a window. Longer time windows require more data, thus potentially slowing down the real-time detection time. The model will be more complex, making it potentially challenging for machine learning to discern the relationships between the data points. Conversely, shorter time windows demand fewer data points and improve the real-time detection time but may introduce more false positives or negatives due to a lack of data. We select 0.5 seconds as the time window to handle the data set as this time window provides a balance between real-time detection speed and complexity. Table II provides the distribution of normal and attacked data, showing 85,045 normal windows and 102,343 attacked windows, totaling 187,388 windows in our collected dataset.

TABLE II
FLIGHT PLAN DATA COMPOSITION (0.5S AS WINDOW SIZE)

Flight Plan	**Number of Windows**		
	Normal	***Attacked***	***Total***
Straight	7,048	11,569	18,617
Curved	30,655	73,052	103,707
Random	47,342	17,722	65,064
Total	85,045	102,343	187,388

To label a window as normal or attacked, we analyze the distribution of data points contained within the window. Specifically, if T% or more of the data points within the window indicate attacked, the window is labeled as attacked. A large T can lead to missed detections as an attack may have a short duration. Setting a small value for T can result in higher false positives since it will inevitably label some normal data as attacked. The value of T depends on the frequency of a UAV's sensors. A sensor with a faster frequency collects more data points per second. Since we select 5 Hz as the sampling frequency and 0.5 seconds as the time window, there are about 2 data points within each window. In this case, the value of T does not make much difference in the performance of our detection system. We select 50% for the value of T.

C. Feature Extraction

This section primarily examines the relationship among the data collected from the GPS, IMU, magnetometer and barometer sensors to select them as features. By transforming raw sensor data into a condensed and representative feature set, machine learning models can achieve enhanced performance and improved accuracy in learning the relationships between these features. For example, many features, such as velocity or acceleration, can be calculated from a combination of sensor outputs. There are a total of 30 features extracted from sensors.

From the GPS receiver, we collect the raw UAV position information, which is a 3D vector containing the UAV latitude, longitude, and altitude. To better detect the stealthy GPS spoofing attack, we calculate and extract the changes in latitude and longitude, denoted as Δlat and $\Delta long$. These two features significantly contribute to the enhanced performance of our model as they effectively capture the impact of this spoofing attack. We also extract the velocity feature, calculated using WGS84 geodesic distance [24], from the GPS receiver. The IMU consists of two sensors: the gyroscope and the accelerometer. Gyroscope captures two types of data: orientation, which pertains to the UAV's rotational position, and angular velocity, which measures the rotation rate. The accelerometer provides information on the linear velocity and acceleration of the UAV. From the gyroscope, we also extract the yaw, pitch, and roll angles from the orientation quaternion as follows:

$$\text{Yaw}, \psi = arctan2(2 * (g_y * g_w + g_x * g_y), -1 + 2 * (g_w^2 + g_x^2))$$
$$\text{Pitch}, \theta = arcsin(2 * (g_y * g_w - g_z * g_x))$$
$$\text{Roll}, \emptyset = arctan2(2 * (g_z * g_y + g_w * g_x), 1 - 2 * (g_x^2 + g_y^2))$$

the symbols g_x, g_y, g_z, and g_w are the respective x, y, z, and w of the orientation quaternion. From the accelerometer, we calculate and extract the pitch and roll angles as follows:

$$\text{Pitch}, \theta = arcsin(a_x/g)$$
$$\text{Roll}, \emptyset = arctan2(a_y, a_z)$$

where a_x, a_y, and a_z are the respective x, y, and z components of the acceleration, and g is the gravitational constant. The yaw, pitch, and roll angles provide valuable information about the UAV's attitude or the direction the UAV is facing. The yaw angle represents the rotational movement around the yaw axis, which aligns with the left or right direction relative to the direction of the UAV's movement. The pitch angle represents the rotational movement around the pitch axis, which aligns with the forward or backward tilt. The roll angle represents the rotational movement around the roll axis, which aligns perpendicular to the longitudinal axis of the UAV. Extracting the yaw, pitch and roll from different sensors will help alleviate the impact from the sensor inaccuracies. The magnetometer and barometer sensors provide the magnetic field strength and air pressure, respectively, which are also useful information in helping identify the deviation of the UAV flight path change.

D. Model Selection

We examine various machine and deep learning models, including XGBoost, Gradient-boosting decision trees (GBDT), Long-Short Term Memory (LSTM) [25], Bidirectional-LSTM (Bi-LSTM), and Recurrent Neural Network (RNN). XGBoost is selected as it has demonstrated success in previous research on GPS spoofing, further corroborated by our study. In addition, we explore LSTM and RNN-based deep learning models due to their capability to capture temporal dependencies and recall past information. These models are widely employed when dealing with time-series data.

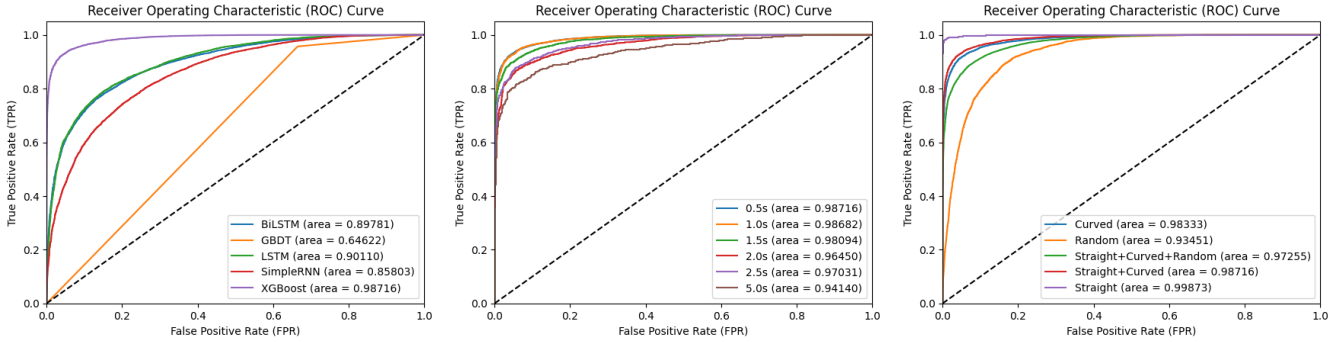

Fig. 5. ROC curves for different models, different time windows and different flight plans

The data set was divided into three parts using a 60-20-20 split, where 60% of the data was allocated for training, 20% for validation, and 20% for testing. We evaluate the models on the testing data. LSTM and RNN-based deep learning models require the data to be normalized to combat the vanishing or exploding gradient problem. Therefore, we apply multiple normalization techniques, such as min-max scaling and quartile range scaling, to the data set. We trained the models for 50 epochs. Gradient boosting models can effectively utilize raw sensor values without data normalization. The max depth of the gradient boosting models was 100, and the evaluation metric was the area under the ROC curve (AUC). The boosting process involved 1,000 rounds, but early stopping was triggered after 50 rounds. After conducting evaluations of these models, we have decided to adopt XGBoost as our chosen model due to its superior performance compared to the other models. The evaluation results are detailed in section IV.

IV. Evaluation

In this section, we present the evaluation results of our proposed sensor-based detection for UAV stealthy GPS spoofing. We use receiver operating characteristic (ROC) curve to visually present our detection performance. The ROC curve shows the tradeoff between the true-positive rate (TPR) and the false-positive rate (FPR) by the area under the ROC curve (AUC). Our detection system is a binary classifier that classifies whether a sequence of UAV sensor data is normal or there is an attack. A perfect classifier would have a TPR of 1.0 and an FPR of 0.0. $TPR = \frac{TP}{TP+FN}$ measures a model's ability to correctly classify an attack. $FPR = \frac{FP}{FP+TN}$ measures a model's performance in incorrectly classifying normal data as attack. The goal is to maximize TPR and minimize FPR.

A. Evaluation Results

In our evaluation, we apply several machine learning algorithms to our data set, including XGBoost, GBDT, LSTM, Bi-LSTM, and RNN. We aim to select the best-performing model that can effectively identify an attack while maintaining a low false positive rate. In addition, we assess the impact of different time windows on the detection performance. Furthermore, our evaluation encompasses the model performance analysis under different flight plans and different onboard sensors.

We first evaluate the performance of the five different machine and deep learning models under the most representative straight and curved flight plans, and present their performance in Fig. 5 where XGBoost gradient boosting model significantly outperforms others, achieving the best AUC of 0.98716 in detecting attacks using the default parameters. Thus, we have chosen XGBoost as the model for our detection system and used it for other evaluations. Table III provides additional performance metrics for each model. Precision (P) $= \frac{TP}{TP+FP}$ measures the percentage of correct positive classifications. Recall (R) $= \frac{TP}{TP+FN}$ measures the proportion of true positive classifications out of the actual positive instances. Accuracy (Acc) $= \frac{TP+TN}{TP+TN+FP+FN}$ measures the percentage of correct classifications. F1 Score (F1) $= \frac{2TP}{2TP+FP+FN}$ measures the harmonic mean of precision and recall. False Positive Rate (FPR) $= \frac{FP}{FP+TN}$ measures the proportion of incorrectly classified positives out of all the true negatives. False Negative Rate (FNR) $= \frac{FN}{FN+TP}$ measures the proportion of incorrectly classified negatives out of all the true positives. In our experiment, since EKF is not able to detect any of our performed stealthy GPS spoofing attack, both TP and FP are 0. The reason EKF accuracy is not 0 is because there are normal data in our collected dataset where EKF will also treat them as normal.

TABLE III
Performance of Machine Learning Models and EKF

Model	Performance Metric					
	P	***R***	***Acc***	***F1***	***FPR***	***FNR***
BiLSTM	0.852	0.910	0.828	0.880	0.357	0.090
GBDT	0.762	0.957	0.764	0.849	0.665	0.043
LSTM	0.852	0.916	0.832	0.883	0.360	0.084
RNN	0.834	0.895	0.804	0.863	0.403	0.105
XGBoost	0.949	0.967	0.941	0.958	0.118	0.033
EKF	N/A	0	0.308	0	0	1

Next, we examine the effects of various time windows on the ROC-AUC performance of our chosen model. We consider only the straight and curved flight plans in this evaluation. We test time windows ranging from 0.5 seconds to 5 seconds. Fig. 5 illustrates the AUC values. The 0.5-second time window performed the best with AUC $= 0.98716$, and the 1-second time window performed similarly well at AUC $= 0.98682$.

The 1-second time window could be a viable choice for the model. However, a shorter time window allows faster detection times. As a result, we select 0.5 second as the time window.

TABLE IV
FLIGHT SCENARIO PERFORMANCE METRICS

Scenario	Performance Metric					
	P	***R***	***Acc***	***F1***	***FPR***	***FNR***
Straight	0.993	0.986	0.987	0.989	0.012	0.014
Curved	0.947	0.964	0.937	0.955	0.127	0.036
Combined (S+C)	0.949	0.967	0.941	0.958	0.118	0.033
Random	0.800	0.708	0.871	0.751	0.067	0.292
All (S+C+R)	0.918	0.912	0.907	0.915	0.099	0.088

Now, we assess the performance of our detection system across the three distinct flight plans with the selected XGBoost model. Fig. 5 illustrates the ROC curve of each flight scenario. The XGBoost model accurately classifies straight flight plan scenarios, where the UAV follows a predictable motion. The straight flight plan is the most common flight scenario, and the model's performance is exceptional, achieving an AUC of 0.99873 and an accuracy of 0.987 shown in Table IV. The FPR and FNR are very low, demonstrating that the detection system is secure and has good usability. The performance slightly diminishes in the case of the combined flight plans, resulting in a lower AUC of 0.97255. Even though the performance drops in the more complex flight scenarios, good accuracies are still obtained with our proposed approach. Fig. 6 shows the performance of each onboard sensor in detecting the stealthy GPS spoofing attack, where we can clearly see that GPS and IMU are the most effective onboard sensors.

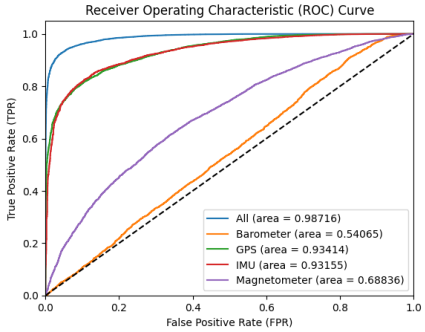

Fig. 6. Performance of each onboard sensor using XGBoost model

V. CONCLUSION

This paper presents a UAV GPS spoofing attack detection system that utilizes the XGBoost algorithm in order to classify normal and attacked UAV sensor data. We collect simulated UAV flight and sensor data through our PX4 and Gazebo based simulation platform, and it is the largest dataset available focusing on UAV stealthy GPS spoofing attack and onboard sensor data-based detection. We examine the impact of different machine learning models, time windows, and flight plans, and identify the best-performing model in all studied scenarios. With the best model, our detection system could achieve a very high accuracy of 98.7% in the most common flight scenario.

ACKNOWLEDGEMENT

This work is supported by the US National Science Foundation awards CNS-2318710 and CNS-2318711.

REFERENCES

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges," *Ieee Access*, vol. 7, pp. 48 572–48 634, 2019.
[2] G. Muchiri and S. Kimathi, "A review of applications and potential applications of uav," in *Proceedings of the Sustainable Research and Innovation Conference*, 2022, pp. 280–283.
[3] S. Z. Khan, M. Mohsin, and W. Iqbal, "On gps spoofing of aerial platforms: A review of threats, challenges, methodologies, and future research directions," *PeerJ Computer Science*, vol. 7, 2021.
[4] Z. Feng, N. Guan, M. Lv, W. Liu, Q. Deng, X. Liu, and W. Yi, "Efficient drone hijacking detection using two-step ga-xgboost," *Journal of Systems Architecture*, vol. 103, p. 101694, 2020.
[5] X. Zhu, T. Hua, F. Yang, G. Tu, and X. Chen, "Global positioning system spoofing detection based on support vector machines," *IET Radar, Sonar & Navigation*, vol. 16, no. 2, pp. 224–237, 2022.
[6] X. Wei, Y. Wang, and C. Sun, "Perdet: Machine-learning-based uav gps spoofing detection using perception data," *Remote Sensing*, vol. 14, no. 19, p. 4925, 2022.
[7] J. Galvan, A. Raja, Y. Li, and J. Yuan, "Sensor data-driven uav anomaly detection using deep learning approach," in *IEEE Military Communications Conference (MILCOM)*. IEEE, 2021, pp. 589–594.
[8] W. Chen, Y. Dong, and Z. Duan, "Manipulating drone position control," in *IEEE Conference on Communications and Network Security*, 2019.
[9] ——, "Accurately redirecting a malicious drone," in *2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2022, pp. 827–834.
[10] A. Raja, M. Jia, and J. Yuan, "Towards the security of ai-enabled uav anomaly detection," in *IEEE International Conference on Communications (ICC)*. IEEE, 2023, pp. 803–808.
[11] S. Wang, J. Wang, C. Su, and X. Ma, "Intelligent detection algorithm against uavs' gps spoofing attack," in *26th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2020, pp. 382–389.
[12] L. Meng, L. Yang, S. Ren, G. Tang, L. Zhang, F. Yang, and W. Yang, "An approach of linear regression-based uav gps spoofing detection," *Wireless Communications and Mobile Computing*, pp. 1–16, 2021.
[13] Z. Feng, N. Guan, M. Lv, W. Liu, Q. Deng, X. Liu, and W. Yi, "Efficient drone hijacking detection using onboard motion sensors," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 1414–1419.
[14] X. Wei, C. Sun, M. Lyu, Q. Song, and Y. Li, "Constdet: Control semantics-based detection for gps spoofing attacks on uavs," *Remote Sensing*, vol. 14, no. 21, p. 5587, 2022.
[15] Y.-H. Sung, S.-J. Park, D.-Y. Kim, and S. Kim, "Gps spoofing detection method for small uavs using 1d convolution neural network," *Sensors*, vol. 22, no. 23, p. 9412, 2022.
[16] E. Basan, A. Basan, A. Nekrasov, C. Fidge, N. Sushkin, and O. Peskova, "Gps-spoofing attack detection technology for uavs based on kullback–leibler divergence," *Drones*, vol. 6, no. 1, p. 8, 2021.
[17] Y. Sun, M. Yu, L. Wang, T. Li, and M. Dong, "A deep-learning-based gps signal spoofing detection method for small uavs," *Drones*, vol. 7, no. 6, p. 370, 2023.
[18] PX4. Px4 user guide using the ecl ekf. [Online]. Available: https://docs.px4.io/main/en/advanced_config/tuning_the_ecl_ekf.html
[19] A. Raja, J. Galvan, Y. Li, and J. Yuan, "Uclp: A novel uav cybersecurity laboratory platform," in *Proceedings of the 22nd Annual Conference on Information Technology Education*, 2021, pp. 23–28.
[20] QGroundControl. [Online]. Available: https://qgroundcontrol.com/
[21] PX4, "Px4 autopilot: Open source autopilot for drones," https://px4.io/.
[22] Gazebo-Classic. [Online]. Available: https://classic.gazebosim.org/
[23] MAVSDK, https://mavsdk.mavlink.io/main/en/index.html.
[24] Geopy. [Online]. Available: https://geopy.readthedocs.io/en/stable/
[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.