Applied GPS Spoofing Detection on Drones

Xiaohan Wang
Eindhoven University of Technology
x.wang7@student.tue.nl

Guangyu Li
Eindhoven University of Technology
g.li@student.tue.nl

Chengqi Liu
Eindhoven University of Technology
c.liu@student.tue.nl

ABSTRACT

Unmanned Aerial Vehicles (UAV), or sometimes known as drones, are increasingly frequently employed in fixed-route scenarios involving tasks such as patrol and transporting goods. However, they suffer from the vulnerability of the civilian GNSS signal, and constantly face the threat of spoofing attacks that could maliciously alter the navigation signal to achieve specific needs of the attacker. Various countermeasures to detect spoofing attacks have been proposed, but as they essentially rely on different measures, there is a lack of research to compare their performance and choice the ideal strategy. In this work, we focus on spoofing detection methods. Specifically, we provided a literature review on current research of detection methods and conducted a comparative study of three selected spoofing detection methods on a single common dataset. We investigated the generalizability of these methods on our dataset, and analyzed their performance, model details, and suitable scenarios. At last, we reflected on the pros and cons of these methods, as well as the weaknesses of our study where the results might not be robust.

KEYWORDS

UAVs, GPS-spoofing detection, GSM, Drones

ACM Reference Format:
Xiaohan Wang, Guangyu Li, and Chengqi Liu. 2024. Applied GPS Spoofing Detection on Drones. In . , 13 pages.

1 INTRODUCTION

Global Navigation Satellite Systems (GNSS) nowadays become the primary techniques to navigate, both for vehicles and aircrafts[17]. The Global Positioning System (GPS) is one of the most famous systems. It is easy to use and can be accessed by public. GPS receivers obtain signals from various satellites, and by estimating the travel time of these signals using precise timing information, they can calculate their distance from each satellite. Using trilateration, GPS receivers can then provide highly precise location information, which is widely used for navigation and positioning.

An Unmanned Aerial Vehicle (UAV) is a flight vehicle that can be controlled either by human or its own flight controlling program. Both of them heavily rely on the navigating system. Due to the low cost of GPS receivers, it is increasingly used to UAV and other vehicles.

However, since the civilian GPS signal is not encrypted, it is vulnerable to spoofing and interference, which can disrupt the navigation and positioning systems of UAVs. By broadcasting counterfeit GPS signals with higher power, an attacker can simulate any position to the UAV. In the worst case, this can result in the UAV being guided to a specific location. For instance, in 2011, Iran captured the advanced US RQ-170 Sentinel UAV by exploiting such vulnerabilities[4]. There are two common types of GPS attacks: GPS jamming and GPS spoofing[13]. GPS jamming involves broadcasting an overwhelming signal that prevents the GPS receiver from receiving the legitimate GPS signal. GPS spoofing involves transmitting false GPS signals to deceive a GPS receiver, leading it to display inaccurate location or time information. In this work, we focus on the detection of GPS spoofing, as there are specific techniques that can identify such attacks using countermeasure algorithms. There are two main categories of GPS spoofing. In the first category, spoofers simply create false GPS signals to deceive the GPS receivers. This method is relatively easy to detect since the received GPS signal is not consistent with what the receivers expect. In the second category, the spoofer first receives the real GPS signal and then fabricates a fake signal based on the authentic one[20]. This type of attack is more dangerous, as it can impose harmful intentions, such as misguiding a drone to a target position. Our work focuses on the second type of attack and constructs the spoofed data based on this scenario.

In this paper, we will first examine the related literature of spoofing detection methods in Section 2. In Section 3, we will apply three detection algorithms to different scenarios. However, these algorithms can be tested on the same dataset using different detection units. Section 4 focuses on the experiments, aiming to verify the validity of the algorithms and produce the insightful results that may benefit the research field. Finally, we will compare the models from different perspectives, and form our own opinions oon the subject.

Our Contribution. Below, we summarize our major contributions in this study:

- We provide a literature review and a detailed analysis of state-of-the-art spoofing detection techniques.
- We propose an experimental framework to ensure that all spoofing detection techniques can be evaluated under the same scenario, and implement four different detection methods under the framework.
- We summarize the differences among the four techniques across various dimensions. From a performance perspective, we evaluate the detection rate using the F1 score, as well as detection time, training time, and detection efficiency. From an algorithmic perspective, we compare the detection unit, applicability for cross-validation, and the tuning threshold. In terms of scenarios, we assess the regularity of the flight path and whether a fixed route is required.

2 BACKGROUND AND RELEVANT WORK

In this section, we conduct a detailed literature review to fill in the gaps of the relevant literature on GNSS, GPS spoofing and spoofing detection, as well as elaborating on our focal models.

First, the section reviews the necessary background information that is required to understand the problem formulation of the spoofing detection tasks, and makes a detailed assessment on the specific nature of threats posed by spoofing attacks on the UAVs. Second, the section investigate the various means of and difficulties in conducting the proposed defences against GPS spoofing, and proposes our categorization of the prevalent spoof detection methods.

2.1 UAVs / GPS Spoofing Attacks

Before diving straight into the detection methods, it is important to familiarize the readers with necessary knowledge about UAVs, GNSS systems, sensors on UAVs, and GPS spoofing attacks. The reason why such knowledge is of great import is that the spoofing detection methods typically will leverage on diverse sensory data on UAVs, and each kind of this data is related to a certain streak of methods.

GNSS. GNSS is an umbrella term that encompasses all satellite-based navigation networks worldwide. Currently, two fully operational GNSS exist: the Global Positioning System (GPS) [17], maintained by the United States, and GLONASS, operated by Russia.

All GPS signal falls into two kinds: military and civilian. The technical implementation difficulty for civilian and military receivers is different in the way that The military signal is encrypted with pseudo-random code [11], thus provide better security defense. The civilian GPS signal, on the other hand, is unencrypted, so the civilian channel can be tampered with and altered to achieve specific objectives.

The GNSS sector of UAVs is able to provide positional information such as latitude, longitude and altitude for navigation of the vehicle. Apart from the GNSS receivers, modern UAVs are often charecteristic of other sensory devices, such as the Inertial Motion Unit (IMU), which logs angular velocity data and acceleration of the vehicle for preset time intervals.

GPS Attacks. Due to the very nature of the civilina GPS signals, UAVs that rely on GPS for navigation are exposed to vulnerability to external attacks due to disclosure of navigation signal format and data format [11]. There are two common types of attacks. The first is GPS jamming, which overload the victim's receiver device with high power signal that forces the target to malfunction [9]. When it comes to UAV formations, a group of orchestrated UAVs set for carrying out specific tasks, the jamming attack disrupts the mutual communications.

GPS spoofing attack is more dangerous than jamming, and is also the focus of our research. It is a common method for injecting false GPS signals into a UAV. By transmitting a faked GPS signal with greater signal power, the adversary can manipulate the UAV's navigation. This tactic enables attackers to redirect the UAV to unsafe areas, alter its flight path, and potentially cause collisions or achieve other malicious objectives [9].

2.2 Taxonomy on Spoofing Detection Methods

A number of taxonomies on spoofing defense have been proposed to provide a systematic view on the question. Generally, spoofing defence involves a two-stage strategy: detecting the attack, and recover or rebuild the trustworthy location and time for subsequent navigation [11]. With respect to our research question, we focus mainly on the first stage of the strategy.

In the work of Schmidt et al. [16], they have proposed a basic classification of current spoofing countermeansures based on the types of data sources, namely signal, cryptography, correlation with other timing sources and radio spectrum & antenna data. This taxonomy is foundational and provide the skeleton of the categorical analysis on spoofing detection approaches. A more recent, detailed taxonomy is proposed in the work [11] on the basis of the previous one that includes slightly more newer information and technology. We adopt their framework in this section and try to fit the methods we investigated throughout the duration of the Seminar into this taxonomy to provide the readers with more systematic overview of the methods.

Next, we will try to provide a brief synopsis of these categories of detection methods, and the corresponding methods we investigated during the Seminar.

- **Signal-processing-based methods** are based on the fact that a GNSS navigation signal is in essence an analog sinusoidal wave at a bounded frequency at the lowest level [16]. It is thus possible to model the signal patterns for anomaly detection task formulation. Basan et al. [3] proposed a normalization technique to identify UAV signal for signs of spoofing attacks, by defining sets of parameters that can highlight an attack and a new database format to support intrusion detection. Oligeri et al. [14] leveraged on the broadcast signals sent by the cellular network infrastructure to verify the position received by the GPS infrastructure for subsequent spoofing attack detection.
- **Encryption-based methods** leverages on extra encryption to create unpredictable transmission signal that is difficult to approximate for attackers. It is suggested that encrypting the signal extension code with a symmetric key in a civilian GPS receiver boosts security significantly Humphreys [6]. However, none of the papers we investigated during the course actually falls into this method class, due to its relative higher requirement for mathematics and cryptography.
- **Signal/geographical-location-based methods** keep track of the direction of arrival of the signal with respect to the received beat carrier phase. This type of methods usually require data received on multiple antennas and other types of receiver data. However, we came across no methods that fall into this category during our Seminar.
- **Drift-monitoring methods** leverages on receiver clocks, IMU or other motional sensor to detect abnormal changes in receiver position or clock. Excessive drift that exceeds certain threshold would be classified as anomaly that will trigger a deception alarm. A considerable number of methods we investigated fall into this category, as error monitoring tasks are computationally efficient for on-device detection systems [21]. Wang et al. [20] designed a Long Short-Term

Memory (LSTM) network based machine learning detection algorithm for path prediction, which could subsequently be used together with IMU based path prediction to monitor the error between UAV's received GPS positions and the predicted ones, triggering detection alarm when both error tests fail. Whelan et al. [21] performed a novelty detection based approach to fitting the error with one-class classifiers after applying dimension reduction on various sensory attributes. Truong et al. [18] considered the behavior of clock bias data when UAVs are under spoofing attacks. They carried out computational emulation to characterize the capacity of detecting spoofing attacks with clock bias data.

- **Mixed methods** tend to combine multiple spoofing strategies instead of using a single strategy for detection. This trend is drawing increasing attention from the researchers as the mixed strategies provide better defense and can be adapted easily for more complicated real-life needs. Michieletto et al. [12], for example, focus on a mixed strategy of GNSS and IMU fusion, IMU-based position estimation and inter-device multilateration to detect spoofing attacks in UAV formations, and design a navigation mode provide stable navigation accordingly if spoofing is detected.

3 METHODOLOGY

3.1 Formulation of the Question

In this section, we introduce the scenario and symbols used in our work. The following notations will be used throughout all the proposed methods for spoof detection.

- P_{GPS} : The received GPS location
- P_{est} : The estimated location by GSM signal
- x_i : The received signal strength by i_th base station
- t : The time(ms) on each data point
- err : The distance between p_{GPS} and P_{est}
- th : The threshold for err
- $\hat{t}$: The duration threshold
- v_t : The drone velocity at time t
- a_t : The drone accelerate at time t
- A_t : A Boolean variable indicating whether the drone is under attack
- S_t : Accumulated error
- b_t : The weight subtracted in each round for S_t

In our work, the received GPS location P_{GPS} can either be the real location of the UAV or a fake location due to spoofing. We employ two approaches to verify the correctness of the location.

P_{GPS} can be used as input to a machine learning model, which acts as a binary classifier. The model outputs a binary label indicating whether the location is spoofed or not. The other approach is to calculate an estimated position P_{est}. By comparing the difference between the two locations and setting an appropriate threshold, the model can determine the accuracy of the GPS signal.

The two statistical methods proposed by Garrett and Gerdes attempt to detect the attacks by directly comparing the error or accumulated error with the threshold [7].

3.2 Model Selection

In our model selection process, the choices of methods are very limited despite a great number of papers we have investigated in the Seminar. For one, this is because the nature of the comparative study puts a restriction on common data sources/features that these methods must share. Dataset coverage is the main practical reason to include only those with shared data types and more importantly, coming with readily available datasets that already include most of such data, so that we do not need to run simulations in the short duration of the Seminar. For the other, also due the timespan of the course, we should balance the reproducibility of the selected methods that do not require transcendental mathematical modelings that are outside the scope of the course. As a result, we selected a set of methods for the most part due to said practical reasons, and these methods are:

3.2.1 GSM Signal Method: Exploit Cellular Network Signal to Estimate Position. When the UAV flies in urban areas, it can receive additional signals such as Wi-Fi and mobile cellular networks like Global System for Mobile Communications (GSM) and Universal Mobile Telecommunications System (UMTS). By receiving these signals along with pre-obtained base station information, such as base station location and cell ID, we can calculate an approximate location P_{est} for the UAV. This is achieved through a weighted estimation of different base stations from the received signal strength (RSS).

We use the exponential distribution to model the RSS, as shown in Equation 1, where x represents the RSS value received by the UAV in dBm units. μ is a parameter that needs to be determined. In our work, we find the optimal value of μ by minimizing the difference between the real GPS position and the estimated position across all traces in the drive-me-not dataset. With the calculated score y, the weight for the corresponding base station can be calculated using the weighted centroid equation 2.

$$y = 1 - f(x, \mu) = 1 - \frac{1}{\mu} e^{-\frac{x}{\mu}} \tag{1}$$

$$w_i = \frac{y_i}{\sum_{i=1}^{N} y_i} \tag{2}$$

With a learned parameter th, we can determine whether a received GPS location is an anomaly or not. However, due to inherent noise, we decide on an attack only after a consistent occurrence of anomalies over a certain period of time $\hat{t}$.

Effectiveness The model can be reliable only under the assumption that the UAV receives signals from a certain number of base stations on most occasions. Figure 1 shows that most data points receive signals from 11 to 15 base stations, which guarantees the effectiveness of the model.

3.2.2 Novelty-Based Method: PCA+One-Class Classifier Based Spoofing Detection. Whelan et al. [21] proposed what they describe as a novelty-based approach to spoofing detection in UAVs based on Principal component analysis (PCA) and one-class classifiers.

PCA is a well-studied dimensionality reduction technique that aims to increase interpretability of large datasets and minimize information loss in the meanwhile [8], by constructing uncorrelated variables that successively maximize variance. In their work,

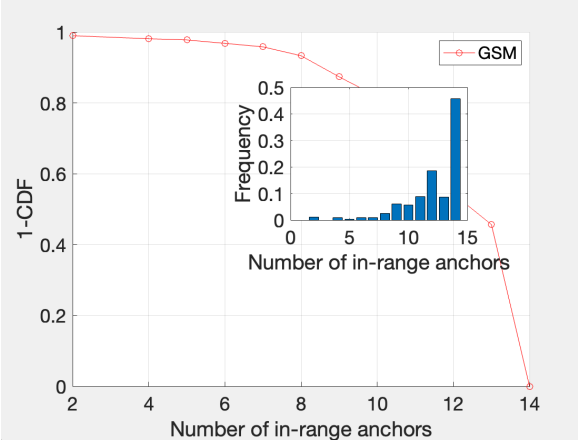

Figure 1: The distribution of received base station signals

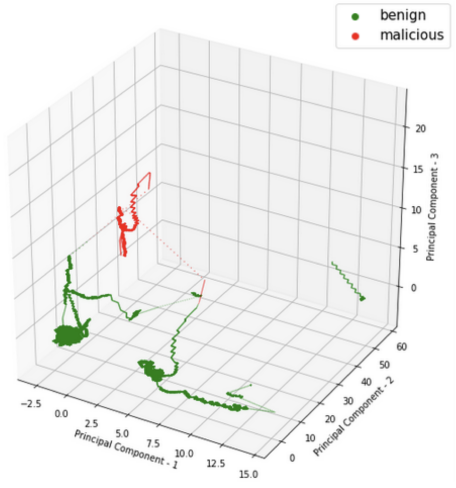

Figure 2: Traces with 85 Features Reduced in 3d Space [21]

Whelan et al. [21] selected 85 relevant features from GNSS, IMU and other sensors, and used PCA to reduce them to 3 components, as is shown in Figure 2.

The reduced principal comopent is then used as the input for One-Class Classification (OCC), where training data is from a single positive class, enabling OCCs to generalize to a representation for recognition of solely positively labels during inference [15]. According to Whelan et al. [21], they conducted the experiment on 3 OCCs, namely, One-class SVM [10], Local Outlier Factors [2], and Autoencoder [19]. One-class SVM (OCSVM) and Local outlier factors are classic anomaly detection machine learning models that are trained in an unsurpervised manner solely on benign datasets. The trained dataset will then be able to tell novelties from the test data if abnormal features are detected. This is especially suitable for spoofing detection tasks that do have spoofed data for training as the training takes only normal data. The Autoencoder here is trained specifically for novelty detection task, where learned detectors is expected to reconstruct the inputs through the hidden feature vector. The idea is that since it is also trained only on benign data, the model can only reconstruct unspoofed inputs and will produce large mean square error when the inputs are spoofed. Hence it is possible to detect spoofed traces by selecting a suitable detection threshold th.

3.2.3 Statistics-Based Method. Since the two methods proposed by Garrett and Gerdes are traditional statistical methods, there is no model selection step [7]. In the data preprocessing stage, only the GSM Signal Method is applied to obtain the estimated position P_{est}, and then err is calculated by haversine for subsequent analysis.

Threshold Method. The method attempts to detect the error err between the drone's position estimated by the cellular base station P_{est} and the position provided by the GPS signal P_{GPS} through haversine to determine if the error exceeds a certain threshold th. If the error exceeds the threshold for multiple times, an alarm is triggered [7]. The formula is defined as

$$A_t = \begin{cases} 1, & \text{if } err > th \\ 0, & \text{if } err \leq th \end{cases} \tag{3}$$

Burst length is defined as the number of times the error exceeds the threshold continuously. The alarm is triggered only when the number of times is more than Burst length. The false positive rate can be significantly reduced through this method [13].

CUSUM Method. The method attempts to calculate the cumulative sum S_t of the errors err and subtracts a weight b_t from the sum at each round. It can achieve more powerful detection and has better robustness than only using threshold methods on a single timestep [7]. The formula is defined as

$$A_t = \begin{cases} 1, & \text{if } S_{t-1} > th \\ S_t = \max\{0, S_{t-1} + err - b_t\}, & \text{if } S_{t-1} \leq th \end{cases}. \tag{4}$$

4 EXPERIMENTS

In this section, we conducted empirical study on the the aforementioned methods based on GSM, novelty-detection, and statistics, using Python under the assumption of our proposed spoofing attack scenario to evaluate their performance, vadility and generalizability. We will first introduce our experiment settings, and then we will present the results of three detection methods respectively.

4.1 Experiment Settings

4.1.1 Our Scenario.

Fixed Routes. The dataset was collected in Doha, Qatar. The drone traveled more than 150 kilometers and about 10 hours [5][13]. We used 8 of the traces and added our generated GPS spoofing signals to replace the real GPS signals. This enables the models that require pretraining to work as intended.

Attacker's Behavior. The attacker can not only monitor the drone's location and speed, but can also launch an attack at any time by sending spoofed GPS signals that the drone cannot distinguish. Also, the attacker's goal is to coax the drone to move to a preset target position that the attacker desires.

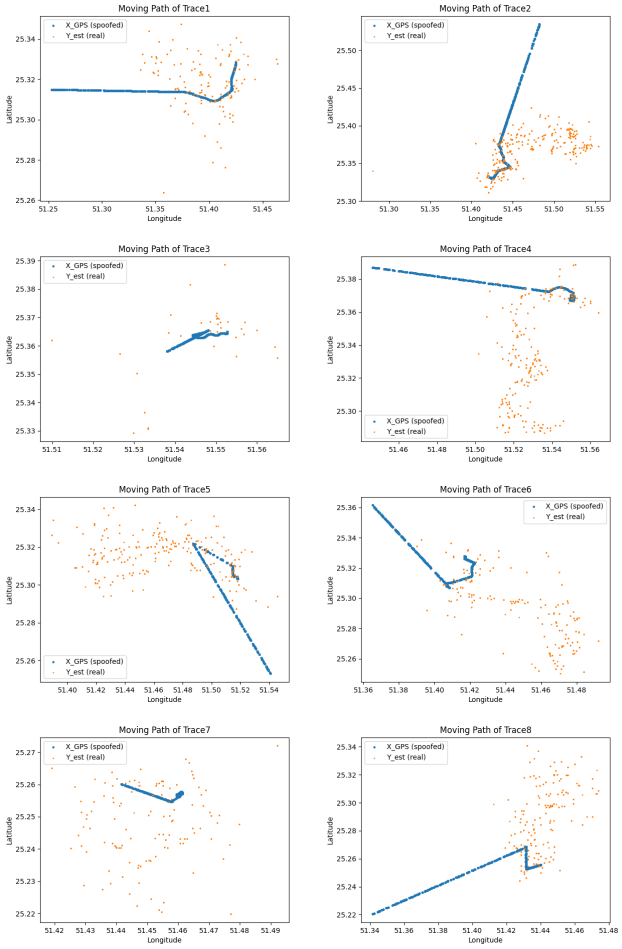

Figure 3: Trace 1-8 in Drive-Me-Not Dataset

4.1.2 Dataset Selection. Since different methods essentially relies on modeling different kinds of sensory and external data that is related to UAVs, there is little to say about the comparative performance of the methods if the dataset we test them on are characteristic of heterogeneous features. Thus, for the convenience of more robust analysis between these methods, all experiments are carried out on one all-inclusive dataset - drive-me-not dataset [5], which comes with the work of GSM method [13] on cellular data. In the dataset we use, the UAV's location, including latitude and longitude, is recorded along a path. This location data is updated every 100 milliseconds. Most flight traces last for 10 minutes, ensuring sufficient data volume for our models. It also contains the cellular signal data used in GSM model, including the base station location, received signal strength and the base station identifier.

However, the dataset lacks some of the critical data fields that powers the nolvety-based methods, such as various kinds of sensory status data from the on-device IMU sensor, and also the actual spoofed traces that are used to verify the above methods. Fortunately, one way around this difficulty is that we could generate such velocity data with the information of the drone's positional coordinates, which makes our experiment possible with the help the generated data.

4.1.3 Generation of IMU Sensory Data. We depended on Haversine formula [1] to calculate the distance of two geographical coordinates. After that, we deemed the average velocity between one timestamp and its next timestamp as the velocity for that timestamp, as the time interval in our dataset is relatively small. In this way, we obtained the velocity and acceleration components of the drone on longitudinal and latitudinal directions. However, this comes with some trouble, as the velocity and acceleration we obtained are not smooth analytically. They appear zigzaggy and chopping, and might change significantly in the next moment, as is shown in Figure 4. A possible explanation for this is that drones could actually be moving irregularly when carrying out tasks. We considered ways to mitigate such effect, such as moving average or interpolation of data points. As much as these methods might soften the velocity curve, they also introduce unnecessary complexity to the generation process. For this concern, we decided to use the unprocessed velocity and acceleration data for our experiment.

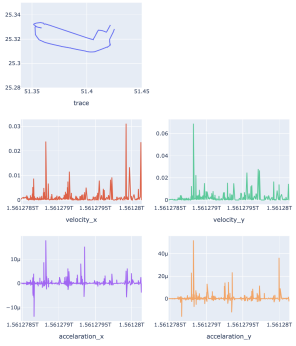

Figure 4: Generation of IMU Sensory Data

4.1.4 Generation of Spoofed Trace. To generate the spoofed trace, a random direction is selected in the middle of benign trace. Then the drone's movement speed is simulated and we generated a series of spoofed data points with uniform motion.

For example, in trace4, the drone traveled from northeast to southwest. The blue dots are the drone's position obtained based on GPS signals, and the orange dots are the drone's position estimated by cell cites. The drone was attacked halfway. The deceptive signal attempted to induce the drone to move at a constant speed to the west.

Description of Drive-me-not dataset. The dataset contains the time, longitude and latitude position, anchor number, CID, LAC, MCC, MNC and dBm data of the drone at every millisecond. It also contains the location and signal information of the GSM cell sites.

4.2 Results

4.2.1 GSM. In the drive-me-not dataset, there are 8 trace data points. We divide the dataset into training and validation sets, each containing 4 trace data points. From the training dataset, we calculate two parameters. The first parameter is the threshold, which

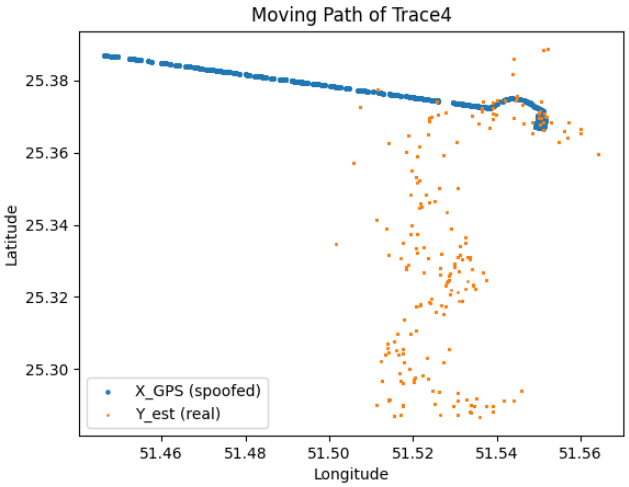

Figure 5: Trace4 in Drive-Me-Not Dataset

determines when the model flags an anomaly if the distance between the estimated position and the received GPS location exceeds this threshold. The second parameter is the duration, which determines when the model identifies a sequence of anomalies as an attack if the duration of anomalies exceeds the specified duration. To achieve a minimal false positive rate, where the model mistakenly identifies benign traces as attacks, we set the maximum duration for sequences of anomalies observed in benign traces within the training dataset. Simultaneously, we aim to minimize the detection duration for actual attacks.

The training dataset consists of traces 5 to 8. According to Figure 6, setting the quantile to 0.74 achieves the minimal detection time and spoof distance. In this scenario, the threshold is set to 0.36 km, and the duration of a sequence of anomalies is 93.3 seconds. These results are summarized in table 1.

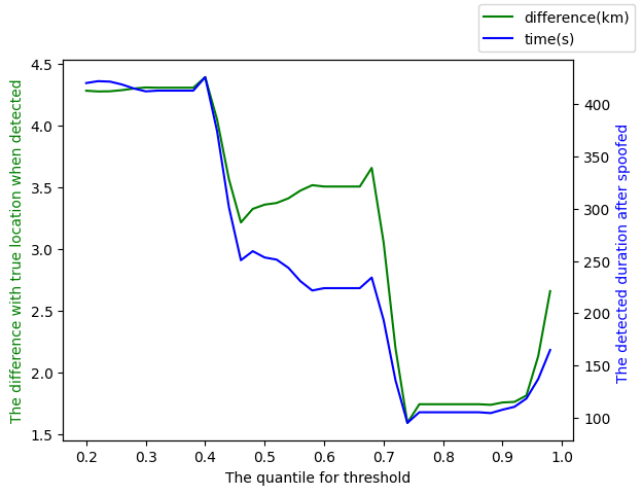

Figure 6: Detection Time and Distance Relative to the Quantile of the Training Dataset

Datasets	False Positive	Duration	Distance
trace1	0.17	100.4	2.23
trace2	0.11	93.3	1.33
trace3	0	93.6	0.94
trace4	0.05	151.0	1.34
trace5	N/A	173.9	1.83
trace6	N/A	96.4	1.57
trace7	N/A	155.1	1.74
trace8	N/A	144.0	1.41

Table 1: Drive me not model

4.2.2 *Novelty-Based Method.*

Data Preparation. With 8 unspoofed traces and their corresponding spoofed traces, the data preparation work for novelty-based method simply used benign traces as the training data, while the spoofed traces were used for testing. Specifically, in the training process of Autoencoder, we additionally split 20% of the training set as validation set for hyper paramter search for the model with the least validation loss.

PCA results. The PCA model is fitted on the entirety of 8 benign traces. After several attempts, the number of PCA component is set to 3 to maximize the performance, whose corresponding cumulative explained variance ratio is 0.95. The input of the PCA model includes the GPS coordinates (the latitude and the longitude), the velocity component on the latitude and the longitude direction, and the acceleration component on the latitude and the longitude direction, as are generated as described above in the experiment settings.

The results of the PCA could be visualized in a 3d space, as is shown in Figure 7, with the each color representing a trace of single points. From the figure, there is little to tell as no obvious pattern could be identified simply through our observation. Next, the PCA results will be used as training dataset for three one-class classifiers.

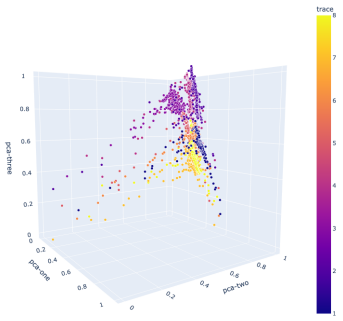

Figure 7: PCA results of All 8 Traces in 3d Space

One-Class SVM Results. The OCSVM model was trained on all benign traces. The ν and γ parameter of the OCVSM need to be

tuned for the best fit, so a parameter search was conducted to optimize for the best ν and γ, with the final value of 0.5421 and 3.6e-3 respectively.

The confusion matrices of the test results on each traces are displayed in Figure 20. It is easy to spot that our OCSVM model indiscriminately labeled all of the test instance as *malicious*. The interpretation of this result cannot be simply reduced to the OCVSM model overfitting the benign dataset, since the test traces also contains close to half the points that are benign. The feature of these benign points would be transformed using the same PCA model we fitted our training set, so they should be easily identified to be normal in theory, which is the contrary of the results we obtained from the experiments.

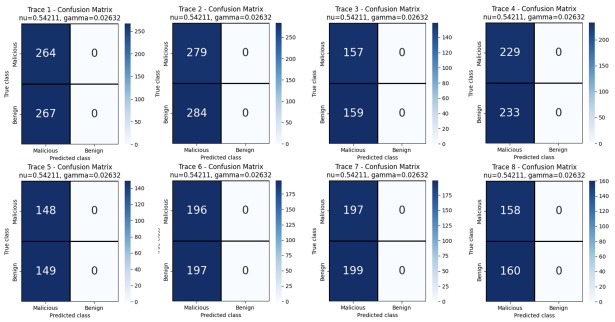

Figure 8: Confusion Matrices for One-class SVM Detection on All 8 Traces

Local Outlier Factor Results. Similarly, the LOF model was also trained on all benign traces. The `num_neighbors` parameter of the LOF need to be tuned for the best fit, so a parameter search was conducted for optimization, with the final values obtained from the search process. However, the values of `num_neighbors` varies per testing trace in practice, so for production environment, this parameter needs to be determined on more context, For example, using the `num_neighbors` with best average score on all testing traces.

The confusion matrices of the test results on each traces are displayed in Figure 9. Again, the one-class classifer did poorly on the testing traces as it indiscriminately labeled all points as *malicious*. Considering the similar result from OCSVM model, it is possible that the generated features did not help OCSVM and LOF methods generalize on *drive-me-not* dataset, since in their original, Whelan et al. [21] applied PCA on up to 85 features, which are far more than what we were able to obtain in this specific experiment. Also, the corresponding F1 score of these two methods are obviously much lower than what Whelan et al. [21] achieved in their original work, for 0.8117 and 0.5893 respectively.

Autoencoder Results. The autoencoder model did produce results that were different from the indiscriminate novelty-labeling of previous two model, but it was not very much ideal, either.

The Autoencoder model was trained using 80% of the training set, and tuned using the rest of the benign data. We performed hyperparameter search in the training process, and the model reached

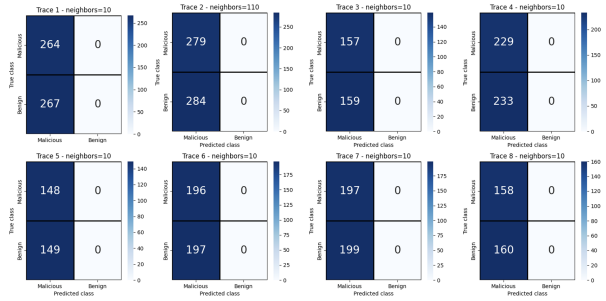

Figure 9: Confusion Matrices for LOF Detection on All 8 Traces

optimal with paramter `hidden_size` = 6, `batch_size` = 32 and `learning_rate` = 1e-3. The model converged very fast, usually within 100 epochs on various combination of hyperparamters, and the overfitting issue was negligible, so no early stopping strategies were applied.

The Autoencoder model works by reconstructing only the benign data, while malicious data cannot be reconstructed. This requires the inputs and outputs of the Autoencoder to be the same when training. After went through the autoencoder, the test traces were reconstructed and then the MSE between the outputs and originals were computed. If the MSE of a data point surpasses a preset threshold T, it is considered as a novelty. The threshold could be tuned with respect to test F1 score on each trace of the eight, as is shown in Figure 10. The expectation is to obtain a general threshold that works on all traces. This could be done by tuning the threshold on the average F1 score of all traces, and the result is shown in Figure 11. At best, our Autoencoder model reaches 0.46 F1 score on average at optimal threshold value. It is slightly better performance than the previous two models, but it is nowhere near the performance of F1 score of 0.9481 achieved in the original work [21].

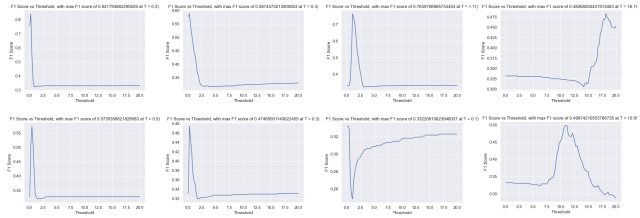

Figure 10: Confusion Matrices for LOF Detection on All 8 Traces

Summary of Novelty-Based Method. The performance of three novelty-based methods is displayed in Table 2. Notices that since OCVSM and LOF models label all testing data as *malicious*, they obtained recall of 1.00 on data with *malicious* labels. However, their overall performance is not satisfactory, compared with that of Autoencoder model.

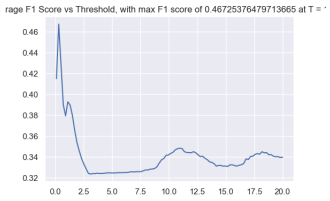

Figure 11: Confusion Matrices for LOF Detection on All 8 Traces

	label	precision	recall	F1 score	Hyperparameters
OC-SVM	benign	n/a	n/a	n/a	
	malicious	0.4968	1.0000	0.6638	nu = 0.5421, gamma = 0.0263
	macro avg	0.2484	0.5000	0.3319	
LOF	benign	n/a	n/a	n/a	
	malicious	0.4968	1.0000	0.6638	num_neighbor varies per test trace
	macro avg	0.2484	0.5000	0.3319	
Autoencoder	macro avg	0.6428	0.6050	0.4808	Tuned via paramter search

Table 2: Performance of three Novelty-based Models

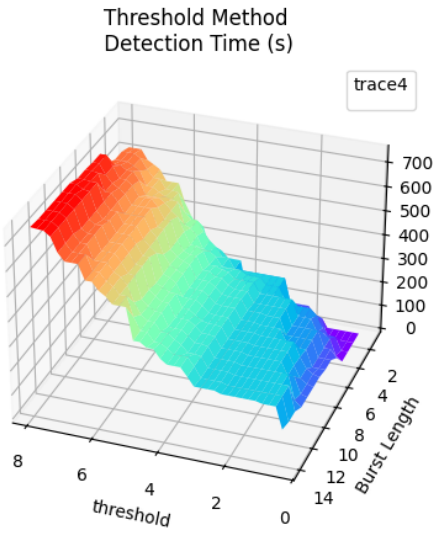

Figure 13: Detection Time of Threshold Method

4.2.3 *Statistics-Based Method.*

Threshold Method. Taking the results on the trace4 dataset as an example, the changes in false positive rate and detection time can be seen under the parameter selections of different thresholds and burst lengths (Figure 12 and 13). This method successfully detected the attacks on all of the 8 traces.

An obvious trade-off between the false positive rate and detection time can be seen from the results: when the burst length and threshold are set to large values, the false alarm rate is close to zero, but the detection time is long, and vice versa.

CUSUM Method. Taking the results on the trace4 dataset as an example, the changes in false positive rate and detection time can be seen under the parameter selections of different thresholds and weights (Figure 14 and 15). This method also successfully detected the attacks on all of the 8 traces.

A similar trade-off can be seen as the Threshold Method: when the threshold and weight are large, the false positive rate is close to zero, but the detection time is long, and vice versa.

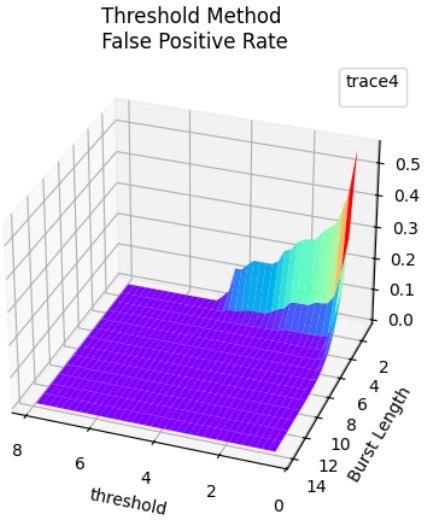

Figure 12: False Positive Rate of Threshold Method

Figure 14: False Positive Rate of CUSUM Method

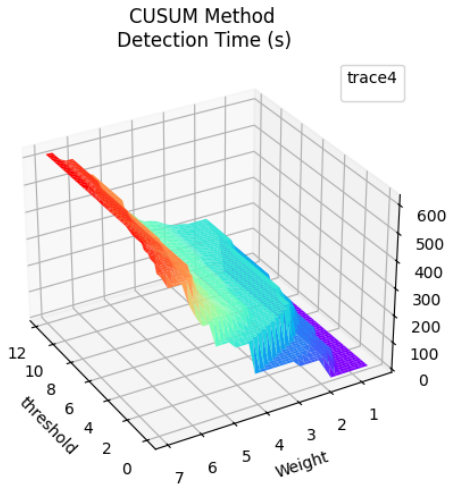

Figure 15: Detection Time of CUSUM Method

Tradeoff. To make this trade-off, we computeD the arithmetic mean of the false positive rate and normalized detection time, and then selectED the minimum point to obtain a recommended parameter setting.

In Figure 16, the recommended parameters of Threshold Method on trace4 is obtained as $th = 0.7$, burst length = 3.

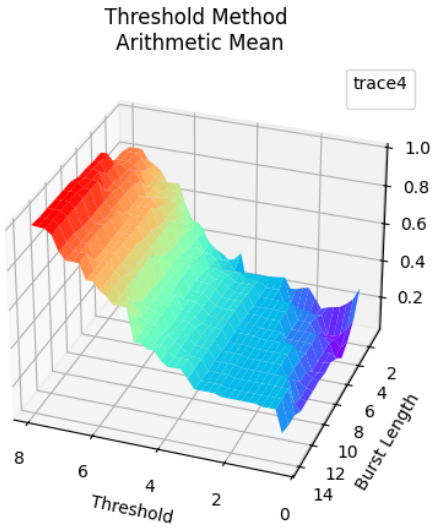

Figure 16: Arithmetic Mean of Threshold Method

In Figure 17, the recommended parameters of CUSUM Method on trace4 is obtained as $th = 0.0$, $b_t = 18$.

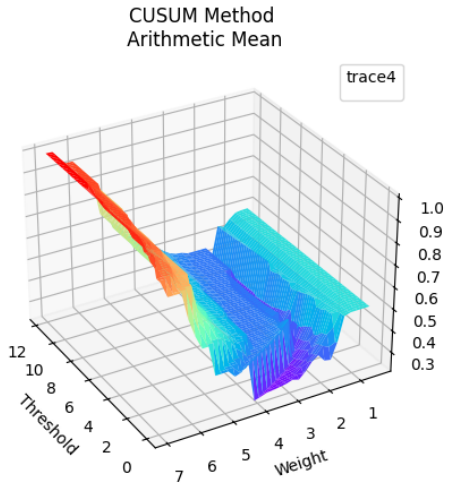

Figure 17: Arithmetic Mean of CUSUM Method

Based on the same method, we obtained the recommended parameter settings on every dataset. However, due to the heterogeneity between different traces, there is considerable variation between the recommended parameters.

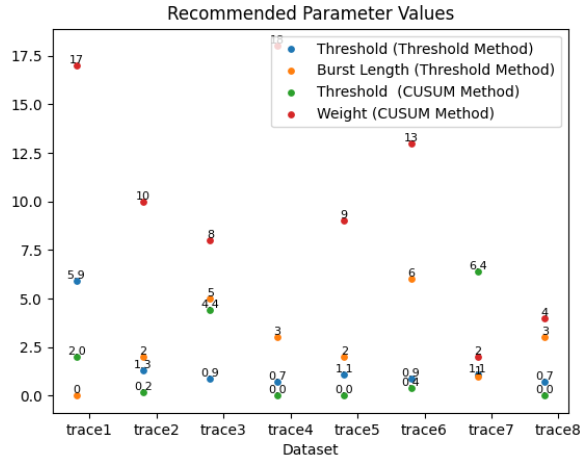

Figure 18: Recommended Parameters on All Datasets

Then, we implemented both methods with the recommended parameters measured on trace4 to test their performance on all of the datasets.

Comparing Figures 19 and 20, it can be seen that the Threshold Method has a higher false positive rate and a shorter detection time, while the CUSUM Method has a lower false positive rate but a quite longer detection time. Therefore, there is also a trade-off between false positive rate and detection time, which in practice should be made based on specific circumstances.

It is worth noting that this rule may change due to the datasets or different parameter settings. The CUSUM Method can also have a higher false positive rate and a shorter detection time than the Threshold Method.

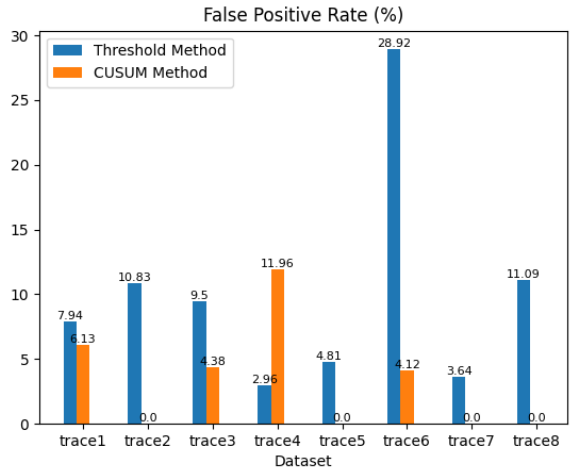

Figure 19: False Positive Rate on All Datasets

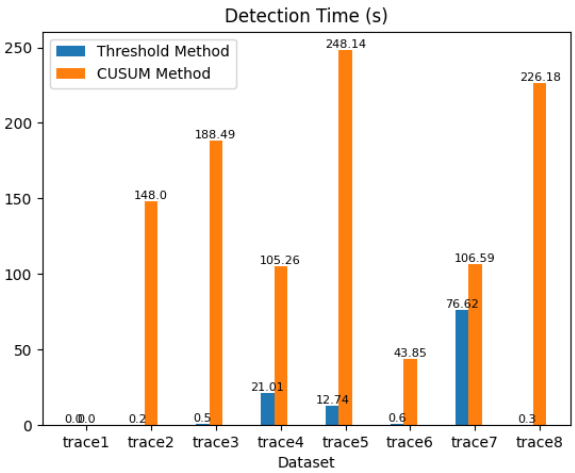

Figure 20: Detection Time on All Datasets

Summary of Statistics-Based Method. Using the recommended parameters measured by trace4, Threshold Method has a higher false positive rate and a shorter detection time than CUSUM Method on our datasets.

Compared with the above two machine learning methods, These two statistical-based methods are with simpler detection principles, fewer parameters and are more lightweight. They do not require a large dataset to get the parameter settings.They are probably more suitable for some constrained small drones.

5 DISCUSSION

5.1 Comparison of Model Performances

Since the number of benign data points in the datasets is much larger than the number of malicious data points, and the number of benign data points can be arbitrarily large depending on the recording time, the true positive rate is not applicable to many methods except PCA+ One-Class Classifier. Similarly, the F1 score is not applicable either.

The best F1 score of all three Novelty-based method OCCs achieved is 0.46, which is a far from what is achieved in the original paper (0.93), and possibly less satisfactory than other two methods. Since the detection model are pretrained and deployed on-device in advance, and due to the relative simplicity of the OCCs, the execution time of the models is considerably fast than the other two. The false positive rate of the GSM model among the test dataset is below 10%, which ensures that the model can correctly identify whether an anomaly sequence is an attack or not.

Although the attack detection rates of all the methods are 100%, this result may not be universal, because we only tested these methods on the eight datasets collected in Doha, Qatar [13].

For the GSM method, since there are lot of noise in the datasets, the detection time is much longer than other machine learning methods.

	GSM	PCA+One-Class Classifier
Performance Metrics		
F1 Score	N/A benign sequences ≫ attack sequence	0.46 (Autoencoder, average of all traces)
False Positive Rate	10.5% (test set)	34.69% (Autotencoder on trace 4)
Detection Rate	8/8 (all traces)	8/8 (all traces)
Detection Time (Quantitative)	98.330s on train dataset	0.0437s (trace 4)
Detection Time (Qualitative)	Relatively Slow	Very fast

	Threshold Method	CUSUM Method
Performance Metrics		
F1 Score	N/A (TP is not applicable)	N/A (TP is not applicable)
False Positive Rate	2.96% (trace 4)	11.96% (trace 4)
Detection Rate	8/8 (all traces)	8/8 (all traces)
Detection Time (Quantitative)	21.01s (trace 4)	105.26s (trace 4)
Detection Time (Qualitative)	Very fast	Slow

Table 3: Comparison of Model Performances

5.2 Comparison of Model Algorithms

The most significant trait of Novelty-based method is that the model inputs and outputs are with respect to a single timestamp in a sequence of trace, instead of sequence per se or a moving window of the sequence.

In general, there is no significant difference in model algorithms among these four methods. Except for GSM, where the model prediction unit is a sequence and the detection unit is a trace, the model prediction unit and detection unit of all the other methods are a single data point.

5.3 Comparison of Model Scenarios

Only the novelty-based method requires fixed route for model fitting process, though this is not a demanding condition as for drones carrying out patrol and transporting missions this is a common scenario. The method is considered very light-weight and easy to deploy, as the training process is very fast and the fitted models are very portable and cost-efficient in terms of on-device computation.

	GSM	PCA+One-Class Classifier
Algorithm-related Aspects		
Unit of Model Prediction	Sequence	Single point in one trace
Unit of Detection	Trace	Single point in one trace
Deviation moment detecting	Can detect	Can detect under more context
Scoring measures	Prediction anomaly sequence in one trace	Predictions of points in one trace
Cross-validation	Applicable	Applicable
Threshold Tuning	Need for evaluating the anomaly sequence	Need for Autoencoder Not need for LOF/OCSVM
	Threshold Method	CUSUM Method
Algorithm-related Aspects		
Unit of Model Prediction	Single point in one trace	Single point in one trace
Unit of Detection	Single point in one trace	Single points in one trace
Deviation moment detecting	Can detect	Can detect
Scoring measures	Predictions of points in one trace	Predictions of points in one trace
Cross-validation	Applicable	Applicable
Threshold Tuning	Need to be tested in advance to obtain recommended parameters	Need to be tested in advance to obtain recommended parameters

Table 4: Comparison of Model Algorithms

In terms of detection efficiency, PCA+ One-Class Classifier and Threshold Method are higher. In terms of training time, CUSUM Method has the smallest theoretical time complexity $O(n)$.

	GSM	PCA+One-Class Classifier
Scenario-related Aspects		
Regularity of Flight path	Not Required	Required
Fixed-Route	Not Required	Required
Pretraining	Required	Required
Detection Effeciency	High	Very high
Training Time	very fast, $O(mn)$ with n data points, m base stations	Converges very Fast for Autoencoder
	Threshold Method	CUSUM Method
Scenario-related Aspects		
Regularity of Flight path	Not Required	Not Required
Fixed-Route	Not Required	Not Required
Pretraining	Required	Required
Detection Effeciency	Very high	High
Training Time	very fast, $O(mn)$ with n data points, m burst length	very fast, $O(n)$ with n data points

Table 5: Comparison of Model Scenarios

6 REFLECTION

At the first glance, the results of the four methods we carried out in drive-me-not dataset with our generated data seems not close to satisfactory. In the section, we point out that our empirical study is challenged by two major aspects of concerns: the intrinsic weaknesses in the methods we selected for this study, and the dataset availability issue that prevent us from conducting experiments on a dataset that is an accurate approximation of real-case spoofing attack.

6.1 Limitation of the Detection Methods

GSM Method. Since the GSM-based model relies on data from cellular networks, the UAV needs to fly around urban areas where it can acquire **GSM signals** from a sufficient number of base stations to estimate its position accurately.

Additionally, for **parameter estimation**, since the drive-me-not dataset only contains 8 traces, it may not cover all scenarios where different velocities of UAVs could influence these parameters. As a result, parameters obtained through trade-offs may exhibit unstable performance when applied to individual test traces.

The final concern relates to the **modeling of RSS weights**. In our work, we assume using an exponential distribution to simulate the weights for each base station. However, due to the lack of data on the power characteristics of each base station, simply unifying them may not be sufficient.

Novelty-Based Method . There are two concerns for the validity of the novelty-based method introduced by Whelan et al. [21]. One is referred to as the **context-awareness** of point-wise prediction model used together with three one-class classifiers. Since the unit of novelty detection is only restricted to a single timestamp in the sequence of flying trace, the models learned from the training set more or less neglect the contextual information of the time point that the trace carries. This context-agnostic assumption casts doubt on the actual performance of the novelty-based method in real-life spoofing attack scenarios.

The other is regarding the **threshold tuning** issue in Autoencoder model that the original work failed to address. The paper does not provide a framework for choosing the right threshold for the model, which undercuts the scientific rigor that the paper should maintain.

Statistics-Based Method . First, this method is highly dependent on the **cell sites** to estimate the position of the drone P_{est}. If the position estimated by the cell sites deviates greatly from the actual position, the performance of the detection method will be poor. In many rural areas, the density of cellular base stations is relatively sparse, and sometimes the drone can only communicate with only one cell site. At this time, the estimated position is extremely inaccurate or will even jump. This phenomenon is particularly obvious on trace3 and trace7.

Second, it is very difficult to make a **trade-off** and set the optimal parameters. In addition, due to the large heterogeneity between different traces, the optimal parameter configuration on one trace may perform poorly on another trace.

6.2 Dataset Availability

Our empirical study is also weakened by the availability of datasets that are suitable for carrying out the comparative research. The datasets on the Internet is usually very limited. In addition, due to the heterogeneity of different detection methods, many of them rely on special data or signals, so it is very difficult to find a dataset that can adapt to enough different methods. The drive-me-not dataset in our study suffers from two main difficulties that undermine its capacity to validate the feasibility of the methods:

- *Lack of Necessary Sensory Data.* For Novelty-based method, the PCA is ideally fitted on a larger variety of time-series data provided by sensory devices on drones [21]. In our research, we synthesized the velocity component and acceleration component data using the provided GPS positional coordinates, while in reality, these IMU-related data is more *independent* of the such GPS positional coordinate data as they are collected from difference sources. Also, even after the generation of sensory data, the diversity of data fields in our dataset is still considerably lower than what is described in Whelan et al. [21], imposing further hardships on the verification of the Novelty-based method.
- *Impracticability of the Spoofing Scenario.* Our generated spoofed traces also compromise the robustness of our experiments

in that there are easily identifiable spoofed patterns. Due to the short duration of the Seminar, we only considered the spoofed scenario where the received GNSS data appears that the drone is moving in a straight line. This is a significant simplified scenario in terms of modern spoofing attacks, as the advanced attackers would model the real-time position and velocity of the drone to launch a more deceptive attack.

6.3 Future Research

To address the above challenges, we outline some potential research recommendations improve the comparative study in future research.

A Full-Fledged Dataset. In further work, we entertain the the possibility of improve the comparative study with a more comprehensive drone spoofing dataset to examine the viability of more spoofing detection methods. Such datasets may encompass data from a real drone carrying out tasks, and thus not just simulation, and more dedicated measurement logs from on-device equipment for relevant physical sensory status data. Also, it is expected to have spoofed traces with more complexity and effort from the attacker. A full dataset offers a closer approximation of real-life spoofing attack scenarios and will significantly increase the validity of our empirical study, making it easier to identify the pros and cons of the selected methods.

Novelty-Based Method. As Section 6.1 has pointed out, the novelty-based method suffered from a potentially untenable assumption that context-agnostic point-wise detection models could in any case power a satisfactory detection system. In this regard, a possible research direction for the novelty-based method could focus on its actual generalizability and robustness under context-aware circumstances. Additional experiments could be set to verify whether the sequential context of single data point is vital for the success of the detection results.

GSM Method. We assume the exponential distribution feature of RSS to calculate the weight. However, further experiments are needed to test the effectiveness of using this assumption. Additionally, with comprehensive spoofed data, the effectiveness of this method can be more thoroughly examined.

7 CONCLUSION

GPS spoofing attacks pose a serious threat to drones and various autonomous navigation systems, and existing detection methods are difficult to compare due to the huge heterogeneity. We conducted a detailed analysis of mainstream spoofing detection techniques and implemented and tested four detection methods using a unified dataset with eight different traces. We then conducted an exhaustive comparison of the four methods in terms of performance, algorithm, and scenario. The results will help to make trade-offs in practical scenarios and defend against GPS spoofing attacks.

REFERENCES

[1] 2024. Haversine Formula. *Wikipedia* (June 2024).

[2] Omar Alghushairy, Raed Alsini, Terence Soule, and Xiaogang Ma. 2020. A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams. *Big Data and Cognitive Computing* 5, 1 (2020), 1.

[3] Elena Basan, Alexandr Basan, Alexey Nekrasov, Colin Fidge, Evgeny Abramov, and Anatoly Basyuk. 2022. A Data Normalization Technique for Detecting Cyber Attacks on UAVs. *Drones* 6, 9 (2022), 245.

[4] David Cenciotti. 2011. Captured U.S. stealthy drone was hijacked exploiting GPS vulnerability. *The Aviationist* (December 9 2011). https://theaviationist.com/2011/12/09/captured-stealth-drone/

[5] CRI-Lab. 2023. Cri-Lab-Hbku/Gps-Spoofing-Detection-Cellular.

[6] Todd E. Humphreys. 2013. Detection Strategy for Cryptographic GNSS Anti-Spoofing. *IEEE Trans. Aerospace Electron. Systems* 49, 2 (2013), 1073–1090.

[7] Garrett Ian Y. and Gerdes Ryan M. 2020. On the efficacy of model-based attack detectors for unmanned aerial systems. *Second ACM Workshop on Automotive and Aerial Vehicle Security* 1 (2020), 11–14.

[8] Ian T. Jolliffe and Jorge Cadima. 2016. Principal Component Analysis: A Review and Recent Developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374, 2065 (April 2016), 20150202. https://doi.org/10.1098/rsta.2015.0202

[9] Shah Zahid Khan, Mujahid Mohsin, and Waseem Iqbal. 2021. On GPS Spoofing of Aerial Platforms: A Review of Threats, Challenges, Methodologies, and Future Research Directions. *PeerJ Computer Science* 7 (2021), e507.

[10] Larry M. Manevitz and Malik Yousef. 2001. One-Class SVMs for Document Classification. *Journal of machine Learning research* 2, Dec (2001), 139–154.

[11] Lianxiao Meng, Lin Yang, Wu Yang, and Long Zhang. 2022. A Survey of GNSS Spoofing and Anti-Spoofing Technology. *Remote sensing* 14, 19 (2022), 4826.

[12] Giulia Michieletto, Francesco Formaggio, Angelo Cenedese, and Stefano Tomasin. 2022. Robust Localization for Secure Navigation of UAV Formations under GNSS Spoofing Attack. *IEEE Transactions on Automation Science and Engineering* (2022).

[13] Gabriele Oligeri, Savio Sciancalepore, Omar Adel Ibrahim, and Roberto Di Pietro. 2019. 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* (Miami, Florida) *(WiSec '19)*. Association for Computing Machinery, New York, NY, USA, 12–22. https://doi.org/10.1145/3317549.3319719

[14] Gabriele Oligeri, Savio Sciancalepore, Omar Adel Ibrahim, and Roberto Di Pietro. 2022. GPS Spoofing Detection via Crowd-Sourced Information for Connected Vehicles. *Computer Networks* 216 (2022), 109230.

[15] Pramuditha Perera, Poojan Oza, and Vishal M. Patel. 2021. One-Class Classification: A Survey. arXiv:2101.03064 [cs]

[16] Desmond Schmidt, Kenneth Radke, Seyit Camtepe, Ernest Foo, and Michał Ren. 2016. A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures. *Comput. Surveys* 48, 4 (May 2016), 64:1–64:31. https://doi.org/10.1145/2897166

[17] GPS SPS Signal Specification. 1995. Global Positioning System Standard Positioning Service Signal Specification, [S].

[18] Victor Truong, Alexandre Vervisch-Picois, Jose Rubio Hernan, and Nel Samama. 2023. Characterization of the Ability of Low-Cost GNSS Receiver to Detect Spoofing Using Clock Bias. *Sensors* 23, 5 (2023), 2735.

[19] Michael Tschannen, Olivier Bachem, and Mario Lucic. 2018. Recent Advances in Autoencoder-Based Representation Learning. arXiv:1812.05069 [cs, stat]

[20] Shenqing Wang, Jiang Wang, Chunhua Su, and Xinshu Ma. 2020. Intelligent Detection Algorithm Against UAVs' GPS Spoofing Attack. In *2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*. 382–389. https://doi.org/10.1109/ICPADS51040.2020.00058

[21] Jason Whelan, Thanigajan Sangarapillai, Omar Minawi, Abdulaziz Almehmadi, and Khalil El-Khatib. 2020. Novelty-Based Intrusion Detection of Sensor Attacks on Unmanned Aerial Vehicles. In *Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet '20)*. Association for Computing Machinery, New York, NY, USA, 23–28. https://doi.org/10.1145/3416013.3426446