CSFC Department
Capstone Project 1: UPM-CSFC-SEM1-2023-18
June 2023

Malicious Behavior Detection in UAVs Network Using Machine Learning

Malak Sabouni 3910004

Wajiha Masood 4010242

Sarah Albadrani 3910158

Zainab Badawi 3820004

Dept. of Cyber Security and Forensic Computing
Faculty of Computer Science and Information Technology
University of Prince Mugrin, Al Madinah Al Munawara KSA

This project report is submitted to the Department of Cyber Security and Forensic Computing at University of Prince Muqrin in partial fulfillment of the requirements for the course Capstone Project I.

Authors:

1- Malak Sabouni

Email: 3910004@upm.edu.sa

2- Wajiha Masood

Email: 4010242@upm.edu.sa

3- Sarah Albadrani

Email: 3910158@upm.edu.sa

4- Zainab Badawi

Email: 3820004@upm.edu.sa

University supervisor:

Naveed Ahmad

Department of Cyber Security and Forensic Computing
University of Prince Mugrin
Address: Medina, Saudi Arabia
Email: N.ahmad@upm.edu.sa

Co-supervisor:

Mohammad Al-khatib

Department of Cyber Security and Forensic Computing
University of Prince Mugrin
Address: Medina, Saudi Arabia
Email: M.al-khatib@upm.edu.sa

ABSTRACT

Unmanned Aerial Vehicles (UAVs) are valuable in a variety of fields, notably industrial control systems monitoring, law enforcement agencies, and military operations. UAVs confront a diverse threat environment due to their dependence on wireless technologies and are vulnerable to threats such as unauthorized access to the UAV's control systems, jamming, and others that can result in events like data breaches and loss of control. To guarantee the security and dependability of UAV operations the UAVs must be equipped with an intelligent intrusion detection system (IDS). It is because traditional IDSs are unsuitable for UAVs as they have limited resources and dynamic operating environments. To address this challenge, we propose in this research a two-stage classification approach for intrusion detection in UAVs. In the first stage, a machine learning model is trained to identify a subset of the data as either normal network traffic or potentially malicious traffic. Whereas the second stage involves training a second machine learning model on the subset of data identified as potentially malicious in the first stage to classify the anomaly. Additionally, we will use feature selection to make this IDS lightweight and more suitable for UAVs. By carefully selecting the most relevant and informative features, we will build a more efficient and effective model that uses fewer resources and is better suited to the dynamic operating environment of UAVs.

Keywords: IDS, UAVs, Machine Learning (ML), Two-stage classification

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

1. INTRODUCTION

1.1 Problem Area

In recent years, there has been a notable surge in the fascination surrounding unmanned aerial vehicles (UAVs). These remarkable machines, commonly referred to as drones, are soaring in popularity due to their diverse range of applications, including but not limited to industrial control systems, monitoring, law enforcement agencies, military activities, and other critical applications. Additionally, now we see UAVs used for many civil and commercial applications, such as package delivery, remote sensing, photography, drone light show, search and rescue, disaster relief, and many more. All these applications demonstrate that security and privacy are of utmost importance and must not be neglected in modern UAVs. However, we find that drones, like many hardware devices, suffer from a variety of vulnerabilities due to the limited computational resources and the focus on other functionalities like performance and usability. This makes many widely used UAVs lack proper implementation of security measures that guarantee the fulfillment of the CIA triad (Confidentiality, Integrity, and Availability), and the recurrent reported drone incidents around the world every week confirm this claim [1]. There are a variety of threats on UAVs, this includes physical attacks, such as shooting them down or disabling them with a physical object, and cyber-attacks, such as Data theft (Man-In-The-Middle attack), GPS spoofing attacks, and Jamming (Denial of Service (DoS)) attacks. Cyberattacks against UAVs are gaining increasing concern as they can be applied and exploited silently and remotely to harm civilians and UAV owners. To defend against all these cyber threats, the first line of defense needed is the presence of an Intrusion-detection-system (IDS), that monitors network traffic and identifies potential security threats or malicious activities.

1.2 Problem Statement

UAV systems are made up of components that are not resilient in cyberspace most of the time and may be readily hijacked by launching cyber-attacks. Almost all commercial UAVs communicate with their base stations via unencrypted channels, they may be exploited by an attacker who can intercept and read sensitive information the drone communicates with the base station such as images, videos, and flight routes. Because of vulnerable to many attacks, UAVs can be dangerous to use in activities such as rescue operations, package delivery, disaster management, etc. Therefore, to be robust in ever threatening cyberspace, the UAV system should be able to withstand cyber-attacks such as: Wi-Fi attacks, DoS attacks, signal spoofing attacks and capturing

live stream attacks. The primary motivation for this project is to secure the UAVs that are used in important sensitive tasks, ensuring that they are robust and safe to use in the face of ever-present cyber threats.

1.3 Research Questions

(RQ1): Will OCC able to detect unknown zero days attack network pattern with high a level of accuracy?

(RQ2): Can ML help to detect all types of attacks launched against UAV?

(RQ3): Will the feature selection Make the proposed framework lightweight so that it can be deployed in resource constrained UAV's?

1.4 Significance of Work

The use of drones has become increasingly widespread in recent years, with applications that touch many sectors including civil, business, personal use, and military. The growth of the commercial drone industry is expected to be significant in the coming years. A recent study projects that this sector will generate approximately $129.33 billion in revenue by 2025 [2]. The Federal Aviation Administration (FAA) also predicts that the size of the commercial drone industry will have tripled by 2023 [3]. As unmanned aerial vehicles (UAVs) become more readily available for use in various industries, the demand for their use in business applications is likely to increase. UAVs will therefore play a crucial role in our modern society as their popularity among civilians is steadily rising. This wide usage highlights the importance of protecting the drone against any attack that may violate the privacy and safety of users. Moreover, drones have been involved in a growing number of security incidents as they have gained in popularity, that's due to many challenges involving securing UAV devices. One challenge in securing UAVs is that traditional intrusion detection systems (IDSs) are not well-suited for use with them. UAVs have limited resources and operate in dynamic environments, making it difficult to apply traditional IDSs effectively. In addition, UAVs rely on wireless communication networks, which can be vulnerable to attacks, and they may be physically vulnerable to attacks as well. Moreover, UAVs often carry sensitive data that needs to be protected.

The significance of this work lies in the potential to improve the security and safety of unmanned aerial vehicles (UAVs) through the implementation of a novel, effective, and reliable intrusion detection system. With the increasing prevalence of UAVs in various industries, there is a need for robust security measures to protect against potential breaches or attacks.

The use of a two-stage classification model for novelty-based intrusion detection in UAVs represents a significant advancement in this field. This type of model can improve the accuracy and effectiveness of intrusion detection by detecting and classifying all types of attacks on UAVs, including zero-day attacks which can be difficult to detect with traditional methods. The potential applications and benefits of this work are numerous. For example, the use of a two-stage classification model for novelty-based intrusion detection in UAVs could enhance the safety and security of delivery operations by protecting against unauthorized access to package data or delivery routes or could improve the efficiency and effectiveness of military operations by enabling faster and more accurate detection of threats.

Overall, this work has the potential to significantly impact the security and safety of UAVs, and it represents a valuable contribution to the field of intrusion detection.

2. RELATED WORKS

To ensure our published work is comprehensive and up to date, we conducted a focused and inclusive literature review covering the period from 2019 to 2022. This allowed us to take full advantage of all relevant prior research and identify gaps in the current state of knowledge. The results of our literature review are summarized in this chapter.

In his Ph.D. thesis, Zhang. [4] proposed two implementation methods to develop IDS for a fleet of drones. The first implementation integrated a robust observer depending on cybernetics theories. It used spectral analysis of the result of wavelet leader multifractal analysis (MFA) to classify network traffic and identifies a DoS flooding attack. The second implementation of the Intrusion Detection System involves analyzing wavelet leader multifractal and using it with machine learning classification to improve detection accuracy further. This methodology was tested only against MITM attacks, as they are stealthier and riskier. This experiment is tested in a simulation lab (Paparazzi platform), which used real time Air Traffic Management RADAR data with artificial attacks. The accuracy of the resulting model is approximately 90%, with a rate of FP equal to 7.9% and an FN of the rate of 8.3%. Since the development of the machine learning model has not been tested in a real environment but rather in only a simulated environment, there are gaps in this work. Additionally, the current paper was conducted based on an experimental dataset, which is considered a limitation to be improved to a more reliable dataset in further work.

Khoei1 et al. [5], made a comparison of the effectiveness of several supervised and unsupervised models for detecting GPS spoofing attacks against UAVs in their paper. This includes

Regression Decision Tree, Gaussian Naïve Bayes, Linear-Support Vector Machine, Logistic Regression, Artificial Neural Network, and Random Forest, for unsupervised learning models, and K-means clustering algorithm, Principal Component Analysis, as well as Autoencoder for the unsupervised models. The results illustrate that the model showed the best performance and outperforms the other models in terms of all matrices Classification and Regression Decision Tree. Among the unsupervised models, the Autoencoder model performs the most efficiently, while the K-means model has the worst performance across all measures. The Gaussian Naïve Bayes model performs worst for the first four metrics. The Artificial Neural Network model has high accuracy and probability to detect attacks but has a longer time in terms of processing, and prediction, as well as higher memory usage. The dataset used in this research is manually generated by collecting real GPS signals by utilizing software-radio units and simulating the attack signals in the MATLAB environment. The simulations include different levels of advancement in GPS spoofing attacks performed.

Shafique et al. [6] used machine learning algorithms to detect GPS spoofing attacks on UAVs by analyzing the characteristics of GPS signals. They created their own dataset with 10 features namely, jitter, jitter (absolute), jitter (local), jitter (RAP), jitter (ppq5), shimmer, shimmer(local), shimmer(dB), shimmer(apq3) and shimmer (apq5) and used K-fold analysis to improve the accuracy of their models. They measured the performance of their model's using accuracy, precision, recall, and F1-score. They were able to achieve good accuracy.

Majeed et al. [7] delivered an intelligent machine-learning approach that adds security to IoT drones. The model employs IoT sensor data, drone information, as well as network data to produce patterns of security statuses and identify security attacks. The research focuses on improving the primary design of drones to address security threats such as data interception and privacy violations. New layers of security are integrated to the traditional drone design to facilitate the performance of security and data analysis mechanisms. This includes establishing a layer for security and privacy to send protected data to the data processing layer and keep it updated with other machine intelligent components. The Naive Bayes model is used for data analysis and threat detection. The model allows data classification into three categories: DOS attacks, jamming, and spoofing. The testing phase of the model was with the KDD'99 dataset, which contains real-time network traffic captured data. The results show that the proposed model resulted in overall accuracy of 96.3%, with class precisions of 96%, 99%, and 93% for the DOS, Jamming, and Spoofing categories, respectively. This indicates that the model can effectively detect security attacks in real-

time. Researchers raised a problem with the algorithm used, Naive Bayes, as it assumes data items to be independent, which may not always be the case. They suggested enhancing their work by using a better algorithm. Furthermore, the use of a generic dataset for training and testing the model is another limitation of this study.

Park et al. [8], In their study, they a deep neural network that applies unsupervised learning as a solution for intrusion detection in unmanned aerial vehicles (UAVs). owing to the fact that no labeling is required for unsupervised learning, which reduces the labeling effort and leads to more accurate results for detecting attacks, this model opted for it. Autoencoder is the algorithm used to build their model. The resulting model is trained with benign flight data and can accurately recognize two sorts of attacks (DoS and GPS spoofing) by utilizing the benign status of data only. The researchers utilized a Hardware-In-The-Loop (HITL) dataset that involves a variety of UAV DoS and GPS spoofing attacks for their experiments, and the dataset contains system logs from simulated flights. Logs were collected under three circumstances: normal flight, DDoS attack, as well as GPS spoofing. The benign data was used for training, while both legitimate and malicious data were used for testing. While the proposed model shows promising results, there are some limitations to consider for real-world deployment. One issue is the computation overhead, as the model is a neural network, which requires more computing resources than individual UAVs can provide. Future studies could address this by reducing the feature vectors size or performing some model compression techniques. Additionally, the model could be improved by training and validating it with actual flight data, rather than simulated data. This would account for aspects such as errors in sensors, climate, and the environment of the communications that can interfere with flight. Finally, the dataset employed in this research is created with the host logs records and not network traffic records, which is a limitation as Network characteristics play a significant role to detect many attacks on the drone including, for example, Man-In-The-middle attacks where it will be stealthier from host logs but noisy from the network aspect of observation, also detecting the attack since it is established in the network instead of detecting it on the host level.

Arregoces et al. [9] conducted a comparative study of one-class classifiers, which are machine learning methods used to detect network intrusions in UAVs. They used the UNSW-NB15 dataset, which includes raw packets of nine different types of attacks, including DoS, Shellcode, Generic, etc. [10]. The training set includes 175341 records and a testing set of 82332 records, which were divided into malicious and benign categories [10]. Next, they evaluated the performance of

these algorithms using relevant criteria for attack detection. They found that one-class support vector machines had the best results among the one-class classifiers they tested.

Aldaej et al. [11], present a modular framework for improving the cybersecurity of drone-based networks using a technique that consists of a combination of logistic regression with random forest machine-learning algorithms to get better accuracy. The framework identifies security threats by analyzing the network data in drones to distinguish different levels of security patterns. The proposed work can detect cyberattacks on in-network data in real-time and has been tested using two generic datasets NS-KDD [12] and KDD Cup 99 [13]. The highest accuracy result with the resulting model was 99.01%. The proposed approach has several limitations, including a lack of focus on physical security threats, potential sensitivity to the quality and quantity of training data, lack of consideration for the cost and feasibility of implementation, and lack of addressing secure data storage and transmission. Future work could address these limitations to improve the robustness and practicality of the envisioned framework.

Park et al. [14]. explored the challenges of communication between unmanned aerial vehicles (UAVs) in a three-dimensional environment where each UAV has its own spatial movement. As the distance between UAVs increases, the likelihood of successful packet transmission decreases, requiring relayed transmission via spatially adjacent UAVs. This complexity makes the communication network between UAVs time-varying. In their research on predicting the success or failure of packet transmissions between UAVs, the authors used Monte Carlo simulation and the Susceptible-Infected-Recovery model to simulate the packet transmission process in a UAV network. They applied basic channel design modeling to study UAV-to-UAV communication and used the spread model of disease on the simulation results. These results were then trained using linear regression and support vector machine techniques to compare the prediction velocities and accuracy of the two systems. This study shows that the nonlinear model of a packet transmission network between twenty UAVs can be accurately trained using machine learning techniques such as linear regression and support vector machine. According to the simulation results, the SVM-QK approach in particular had 0.0000531% root mean square error and was able to predict packet transmission probability more quickly and accurately than linear regression. This method can be used to predict the transmission success or failure of packets in a time-varying UAV network.

In their research, Zhao et al. [15] compared the performance of two types of datasets: a synthetic dataset generated with certain settings, including a design matrix the data will be

transformed by scaling it such that it has a mean of 0 and a standard deviation of 1 and feature costs from a Gaussian distribution, and a real-world dataset with six different types of data and various features related to packet sizes and inter-arrival time. The synthetic dataset was divided into three subsets: a training set, a validation set, and a testing set, while the real-world dataset was divided into three datasets with 54 features and three datasets with 18 features. To analyze their data, they used a set of mathematical statistics as features, including measures of central tendency, spread, symmetry, peakedness, and extreme values. However, there is no widely accepted standard for feature selection in this context, and the authors did not provide any further explanation or description of the features they used. Further research may be needed to develop a more systematic and rigorous method for feature selection.

Pawlak, J. et al. [16] suggested using Machine learning (ML) which can be used to identify and classify jamming attempts against unmanned aerial vehicles (UAVs). Software-defined radio (SDR) is used to conduct four types of attacks: barrage, single-tone, successive-pulse, and protocol-aware jamming. These attacks are performed on a drone using orthogonal frequency division multiplexing (OFDM) communication, and the effects of each type of attack are evaluated in terms of jamming range, complexity, and intensity. SDR is used to record radiometric parameters before and after each attack in controlled testing conditions, and six ML methods are used to enable autonomous jamming detection and classification. Features used in the algorithms include signal-to-noise ratio (SNR), energy threshold, and various OFDM parameters. The algorithms are quantitatively evaluated using metrics such as detection and false alarm rates to enable effective decision-making for improved reception integrity and reliability. The resulting machine learning method is able to detect and classify jamming with an average accuracy of 92.2% and a false alert rate of 1.35%. In the future, this research will include a complexity analysis of the developed classifiers, investigation into additional types of jamming (such as deceptive and reactive), incorporation of maximum-likelihood-based classification and advanced SNR probing, extraction of features for jamming detection and classification at different UAV altitudes, and exploration of UAV-specific anti-jamming solutions (e.g., flight scheduling, path optimization).

Authers & year published	Used dataset for UAV attacks (not generics attacks dataset)	Network-based dataset	use of primary real dataset (not simulated)	Feature analysis and selection tools	Involved using Two class classification	IDS detect any attack including zero days attacks	IDS Classifies the attack type
thesis, Zhang. (2022)	✓	✓	×	×	×	×	✓
Khoei1 et al. (2022)	✓	✓	×	✓	×	×	✓
Shafique et al. (2021)	✓	✓	✓	×	×	×	×
Majeed et al. (2021)	×	✓	×	×	×	×	✓
Park et al. (2020)	×	×	×	✓	×	✓	×
Arregoces et al. (2022)	×	✓	×	✓	×	✓	×
Aldaej et al. (2022)	×	✓	×	×	✓	✓	×
Jinsoo Park et al (2022)	×	✓	×	✓	×	×	✓
Zhao, L. (2018)	✓	✓	✓	×	×	×	×
Pawlak, J. et al. [2021]	✓	✓	✓	✓	×	×	✓
Our proposed system	✓	✓	✓	✓	✓	✓	✓

***Table 1**: Related Work*

3. PRELIMINARIES

3.1 Intrusion Detection System

A Network Intrusion Detection System (IDS) tracks the flow of data across a network in order to detect suspicious activity and alert users when it is detected. An IDS is an essential component of any network security strategy, as its Real-time observation of network traffic and alerts for any unusual activity, any attempts to attack and breach the system which may stop the attack before it starts and assist in containing it before it spreads through the network components. Typically, when a threat is detected, an alert will be sent to the system administrator, or it will be gathered centrally and collected by the SIEM system. In this process, outputs from several sources in the network are integrated with filtering techniques in order to distinguish benign from malicious activity. The idea of automated intrusion detection systems (IDS) was first suggested in 1980 by Jim Anderson, a specialist in information security and member of the U.S. Air Force, in a study titled "Computer Security Threat Monitoring and Surveillance"[17].

3.1.1 Intrusion Detection System Types

Based on where the IDS will be installed and the scope and nature of the monitored activities, intrusion detection systems (IDS) can be classified into two main categories: network-based intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS).

3.1.1.1 Host-based Intrusion Detection Systems (HIDS)

A HIDS monitors the system logs and activities on a specific machine and is only able to monitor the network traffic on the machine's interface. Generally, Activity monitored includes changes in the file system, network communications and traffic, system logs, user and application behavior, and resource utilization. This type of IDS uses the gathered logs to detect and analyze any abnormal behavior within one host system and alert any suspicious activity to the administrator. A Host-based intrusion detection system (IDS) has advantages and disadvantages compared to other types of IDS. On the one hand, a host-based IDS can provide detailed monitoring of a single computer or host device, which allows it to access the local events on the host and monitor it such as analyzing the decrypted packets and running processes. This aids in detecting intrusions which could potentially go unnoticed by the network-based IDS. On the other hand, Host-based intrusion detection system may not offer the same level of visibility and security as a network-based IDS, which analyzes all traffic across a network because it only monitors a single device. A host-based

IDS can also be resource-intensive and may need advanced technical knowledge to install and maintain [18]. Additionally, one significant limitation of host-based IDS is that it works after the threat intrudes on the system, which contains risk compared to defending the system on the network level.

3.1.1.2 Network-based Intrusion Detection Systems (NIDS)

Network-based intrusion detection systems (NIDSs) monitor network traffic to detect abnormal behavior. The essential function of a network-based IDS is to detect anomalies and patterns in network traffic that indicate security threats. Typically, IDSs of this type are installed on network devices such as routers and switches. Although network-based IDS will not have access to details of the activity on each host as is the case for host-based IDS, it has the advantage of monitoring traffic from the choke point to alert users before the threat intrudes into the hosts. Additionally, a network-based IDS can offer extensive information on the nature and origin of a threat, which can be helpful for locating the attack's origin and implementing proper incident responses.

3.1.2 Intrusion Detection System Detection Approaches

The approach of detection in IDS describes the process followed by an IDS to flag an activity as malicious. There are two main approaches that are commonly used in intrusion detection systems (IDS): signature-based detection and anomaly-based detection.

3.1.2.1 Signature-based Detection

A signature-based intrusion detection system utilizes a collection of previously recorded indicators of attacks to detect malicious traffic. In this approach, incoming network traffic is compared to the stored signatures in the database, and if a match is found, the system will respond with the suitable action, such as blocking the traffic or generating an alert. Using this method has the advantage of accurately identifying known attacks, but it may not be effective at detecting new or previously unknown attacks that are not included in the signature database. Therefore, it is important to regularly update the signature database with new attack patterns to maintain the effectiveness of the system.

3.1.2.2 Anomaly-based Detection

Anomaly-based intrusion detection systems use machine learning algorithms to identify patterns or anomalies in network traffic that may indicate an attack. This approach involves creating a profile of normal network behavior and then flagging any deviations from this profile as potential attacks. This method has the advantage of the ability to detect previously unknown or novel attacks, but it may also produce false positive alarms if the system flags benign traffic as suspicious. Using anomaly-based detection of network traffic, the project proposes an IDS for UAVs that can detect any UAV cyberattack.

3.2 Using Network Based Traffic with Machine Learning

This project presents a machine learning-based intrusion detection system (IDS) that utilizes advanced algorithms to monitor network traffic and classify it as legitimate or malicious. By harnessing the power of machine learning, this IDS enhances the ability of network-based security systems to identify and respond to potential threats. The benefits of incorporating machine learning into intrusion detection systems can be achieved through various methods. One such approach is to use machine learning algorithms and train them with a considerable amount of previous network traffic to identify the pattern and characteristics of malicious traffic. Nevertheless, it is worth considering whether a dataset that was extracted from network traffic could be effectively used to classify malicious traffic from benign traffic. In fact, network traffic records provide a lot of valuable data that can be utilized for analyzing and effectively detecting the presence of an intrusion. For example, Mhawi, et al. created IDS for generic network traffic, and they extracted 37 features from the network traffic that contains information about the packet length, the original and final IP addresses, and the source and destination ports, TCP flags like SYN, ACK, PSH, etc. and Flow Duration that defines the duration that has been taken to complete the communication. With using of these features, they were able to develop an AI-based IDS that has an accuracy of 99% [19].

3.3 Machine Learning

Based on the DATAVERSITY Education [20], Machine learning is an important technique for using artificial intelligence (AI) effectively. While it is often referred to as AI due to its ability to learn and make decisions, it is actually a subset of AI. It has a long history, starting as a stage in the development of AI in the late 1970s and later evolving on its own. Today, machine learning is used in many advanced technologies and is a crucial tool for cloud computing and e-commerce. it is a key component of contemporary business and research for many companies, helping computer

systems improve over time through the use of algorithms and neural network models. These algorithms generate a mathematical model using sample data, also known as training data, without being specifically programmed to do so. They are often developed using frameworks such as TensorFlow and PyTorch.

3.3.1 Types of Machine Learning

The kind of data that data scientists wish to forecast determines the kind of algorithm they use. Machine learning algorithms can be classified based on how they learn to improve their accuracy in making predictions. There are four main types: reinforcement learning, semi-supervised learning, unsupervised learning, and supervised learning. The type of data that data scientists want to predict determines the type of algorithm they use.

3.3.2 Supervised Learning:

In supervised machine learning, data scientists provide algorithms with labeled training data and specify the variables they want the computer to search for correlations between. The input and output for the algorithm are both given. The model adjusts its weights as input data is fed into it until it is properly fitted, which occurs during the cross-validation process to ensure the model is not overfitted or underfitted. Supervised learning is often used by companies to classify spam emails into a separate folder, for example. Techniques used in supervised learning include neural networks, naive Bayes, linear regression, logistic regression, random forest, and support vector machines (SVM).

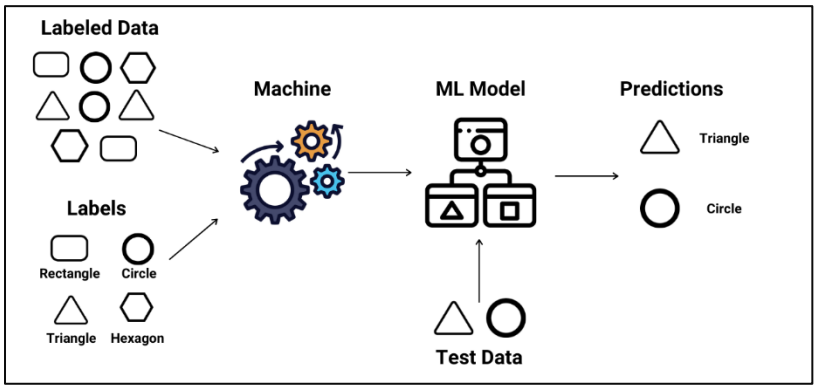

Figure 1: *Supervised learning diagram*

3.3.3 Unsupervised Learning:

In unsupervised machine learning, algorithms are trained on unlabeled data and search for significant relationships within the data sets. The input data and predictions provided by the algorithms are predefined. Unsupervised learning involves analyzing and grouping unlabeled datasets using machine learning techniques to identify hidden patterns or data clusters without human guidance. This approach is useful for exploratory data analysis, cross-selling tactics, consumer segmentation, and image and pattern recognition, as it can identify similarities and differences in the data. Dimensionality reduction is also often used in unsupervised learning to reduce the number of features in a model, with popular methods including singular value decomposition (SVD) and principal component analysis (PCA). Other algorithms used in unsupervised learning include neural networks, k-means clustering, and probabilistic clustering techniques.

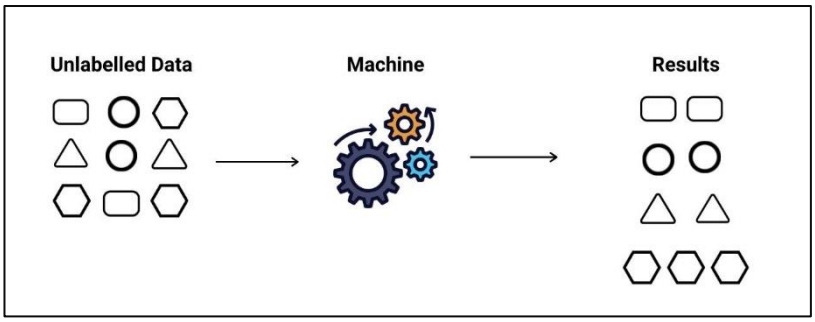

Figure 2: *Unsupervised learning diagram*

3.3.4 Semi-Supervised Learning:

Semi-supervised learning combines the approaches of supervised and unsupervised learning. Data scientists provide the algorithm with mostly labeled training data but allow it to explore the data and draw its own conclusions about the dataset. This method is used in situations where a model must learn and make predictions on new instances using a small number of labeled examples and a large number of unlabeled examples. It is a type of learning problem and algorithms are created to solve it.

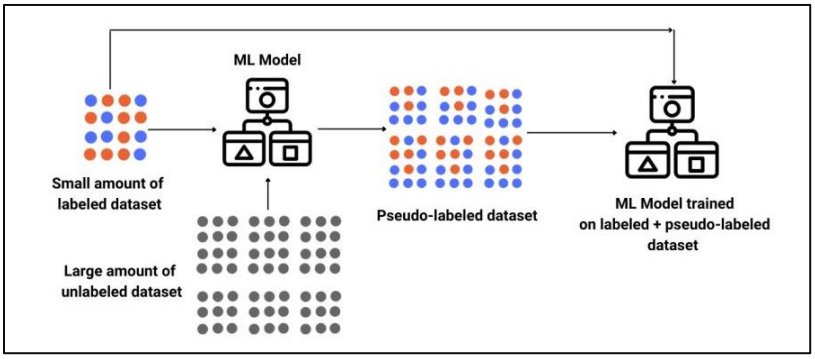

Figure 3: *Semi-Supervised Learning use case*

3.3.5 Reinforcement Learning:

Unlike supervised learning, reinforcement learning does not use sample data to train its algorithms. Instead, it relies on trial and error to learn as it goes along. This type of machine learning is often used to train a system to complete a multi-step process with clear criteria. Data scientists program an algorithm to achieve a specific goal and provide it with positive or negative feedback as it determines how to do so. The algorithm typically makes its own decisions about the course of action to take. Reinforcement learning is based on the idea of reinforcing successful outcomes in order to find the optimal solution or strategy for a given problem.

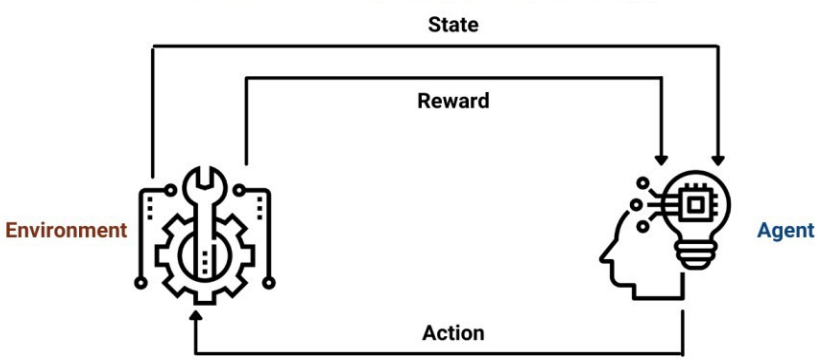

Figure 4: *Reinforcement learning diagram*

3.4 One Class Classification (OCC)

One-class classification is a type of supervised learning that is useful for situations where there is data only for one class, and we want to create a model to classify incoming data points or when the dataset is highly unbalanced, with many data points belonging to one class and a small number belonging to the other. Hence, OCC trains a model based on samples from only one class, assuming that the other class(es) do not exist or are not well-sampled in the training data.

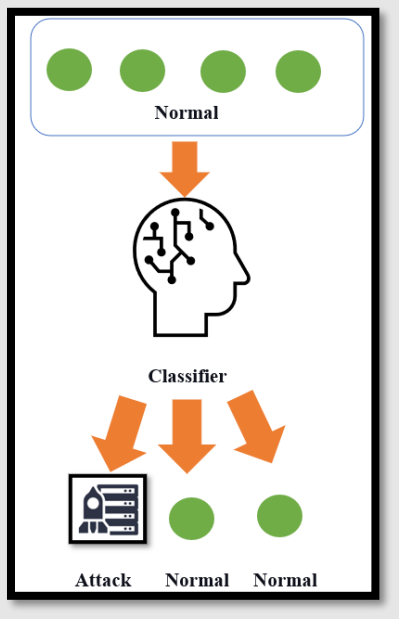

Figure 5: *OCC*

In the figure above, a classifier will learn from the provided samples of one class to detect out-of-class samples. Here, OCC is used to detect attacks. So, classifier is trained to recognize anomalous traffic objects given a dataset of normal traffic object.

3.5 Novelty Detection

Novelty detection is a statistical method utilized to specify unusual data and figure out whether these new data are outliers or inliers relative to the norm. An inlier is a data value that conforms to the general pattern or trend in a dataset. These values can be easily identified because they fit within the expected range or distribution of the data. Outliers, on the other hand, are data values that lie outside the expected pattern or trend. These values can be more challenging to identify because they may be caused by errors in data collection or measurement, or they may represent legitimate but unusual or unexpected observations. Novelty detection can be used in a range of

fields to identify anomalies in routine operations, such as detecting network intrusions, hacking attempts, or jet engine failures.

3.5.1 One-Class Classification Novelty Detection

One-class classification is a machine learning approach that is used for novelty detection, where the goal is to identify instances that are different from the "normal" or "positive" class. In one-class classification, only examples from the positive class are used for training, and the goal is to learn the characteristics of this class in order to identify instances that are different from it. This is particularly useful in situations where obtaining examples of the "negative" class is difficult or expensive, such as in the monitoring of machine behavior. By using a one-class classification approach, it is possible to train a classifier on the positive class and use it to identify instances that are likely to belong to the negative class.

3.5.2 Novelty Detection based on Probability.

Novelty detection based on probability is a machine learning technique that involves identifying events or data points that are unusual or unexpected. If the probability is below a certain threshold, the data point is flagged as being novel or unusual. There are several different methods to novelty detection based on probability, some common methods to determine the probability distribution of the data includes kernel density estimation and Gaussian mixture models. Once the probability density function has been estimated, a novelty threshold can be determined using statistical techniques such as the p-value or the false discovery rate. Probability-based novelty detection can be applied in a range of contexts, including intrusion detection, and quality control. It can help to identify unusual patterns or events that may not be detected using other methods.

3.5.3 Novelty-based Intrusion Detection System for UAVs

A novelty detection system can be used to identify unusual or unexpected events or patterns in the UAV's sensor data or environment, and this information can be used to flag potential instances of malicious behavior. For example, a novelty detection system could be used to identify unusual patterns in the UAV's flight path, such as sudden changes in direction or altitude, or deviations from a predetermined course. It could also be used to detect unusual objects or events in the UAV's environment, such as the presence of unauthorized personnel or vehicles, or the presence of objects or structures that are not typically found in the UAV's operating area. In addition, a novelty detection system could be used to analyze the UAV's sensor data for anomalies that could indicate the

presence of malicious activity. For example, it could be used to detect unusual patterns in the UAV's communication or power usage, or to identify discrepancies in the data being collected by the UAV's sensors. In brief, a novelty detection system can be a useful tool for detecting and preventing malicious behavior in UAVs and can help to ensure the safe and reliable operation of the vehicle.

4. PROPOSED APPROACH

The following chapter will provide a general overview of our proposed approach, as well as delve into the specific steps involved in the development of our machine learning models. These steps include data collection, feature selection, model training and testing, and performance evaluation. Finally, we will present a visual representation of the overall process flow.

4.1 Overview

For the aim of detecting unusual or unexpected behavior in unmanned aerial vehicles (UAVs), our approach proposes the developing of an Intrusion detection system (IDS) based on machine learning. This approach involves the use of a two-stage classification process, where identifying threats on UAVs will pass through two phases of classification.

4.1.1 Stage I

The first stage includes applying One-class classification to detect the existence of any threat. One-class classification can be used as a novelty detection method of machine learning (ML) when we want to identify instances of a particular class that are significantly different from the training data. For example, in the case of unmanned aerial vehicles (UAVs), we want to use one-class classification to identify abnormal UAVs traffic. We can then use this model to classify new UAV observations as either normal or abnormal based on how closely they match the training data. We would first need to gather or generate a dataset containing only normal UAV behavior characteristics, then use this dataset to train a one-class classification model.

Therefore, in this stage, the model will be able to detect any kind of attack because it has been trained only with benign data and this will give an advantage because we can detect any type of attack including zero-day attacks, or attacks may have new unknown characteristics. A zero-day attack is a type of attack that aims to exploit a previously unknown vulnerability in a software or system. These attacks can be difficult to detect as they take advantage of a vulnerability that was not previously known or disclosed, therefore there are no existing defenses or patches to prevent the attack from occurring. In the proposed approach, the first stage classification overcomes this

limitation as anything not benign will be considered malicious behavior. If an observation is classified as abnormal, it may indicate the presence of a novel UAV behavior that warrants further investigation, and the second stage will help to figure out the attack type occurred.

4.1.2 Stage II

In the second stage, it will use the same traffic, but with the purpose of specifying the type of attacks. In this phase, the model will be trained with dataset that contains several types of attacks characteristics, which allows it to categorize it into labeled types of malicious behaviors. Either to known attack names which includes DDoS attack, GPS spoofing and Man-In-The-Middle attacks, or it will be classified as unknown unexpected behavior, which will be usually a zero-day attack, or an attack that was not recorded or trained before in the AI model.

In brief, the key contribution of our approach is to develop an IDS for UAVs that is designed to detect a wide range of attacks, including zero-day attacks, which can be particularly dangerous for UAV systems. By using one classification as a novelty detection method and training the algorithm on benign data, we can alert the presence of any type of attack. Thereafter in the second stage-classification, using supervised learning algorithms that is feed with different attacks data, the IDS will classify malicious behavior in UAV systems then into what type of attack it is, even if it is completely new and unknown behaviors.

4.2 Data Collection

The quality and diversity of the data used to train a machine learning model are crucial for its effectiveness and accuracy. Therefore, it is important to carefully gather and process a large and varied dataset during the data collection phase, which is the first and most important step in building the model. Failing to do so can negatively impact the model's performance. Our project will consider multiple options to gather our dataset in order to maximize the accuracy level. We have found multiple available datasets, one of them was generic network traffic to classify benign and malicious traffic [21], while others were related to drones such as VTO lab dataset [22], and another related dataset was UAV Intrusion Detection Dataset that created by Zhao from George Mason University [23], additionally, there is one interesting UAV attacks dataset that provided by IEEE [24] and contains drone different malicious and benign records. However, all mentioned datasets were extracted from the drone system log, which will work for host-based IDS, while in our proposed IDS a network-oriented dataset is preferable, as it works on the network level where almost all attacks possible on the drone are centralized around it. As previously mentioned, a publicly available

dataset could be used in building our model, however, it is possible also to create the dataset, first by capturing the network traffic using one traffic sniffer like TCPDump or Wireshark, then will extract its features and use it to feed the model with benign and malicious types of traffic. To collect data for the second phase of classification, we can simulate various types of attacks on the drone and record the resulting traffic. For training and testing a machine learning model for drone intrusion detection, can use either actual traffic data or simulated traffic data. Both options have their own advantages and can be useful in different scenarios. Using real-time traffic may provide more accurate results, as it represents actual conditions and scenarios the model may encounter in practice. However, virtual traffic can also be helpful in generating a larger and more varied dataset, and for testing and evaluating the model under different conditions. Ultimately, the choice of whether to use real or virtual traffic will depend on the resources available for our project. Using a virtual lab to mimic the drone traffic to generate more different virtual traffic is possibly more, as the real resource we have is humble and simple and a virtual lab environment like the open-source platform paparazzi [25] will increase the size and variability of drone traffic records. In brief, the careful selection and use of appropriate datasets is a crucial aspect of building a successful machine-learning model for our drone intrusion detection system. The goal is to select the most suitable datasets in order to create a rich and diverse set of data that will enable training and improving the accuracy of the model. Doing so, can ensure that the proposed system for detecting and responding to various types of drone attacks is effective and reliable. To achieve this, will consider multiple options and choose the ones that are most appropriate for the needs.

4.3 Feature Selection

Features are characteristics or qualities of a dataset that can be used to describe the data and identify patterns, trends, and relationships. In data analysis, features are often selected and extracted from a larger dataset in order to facilitate the analysis and make it more manageable. In the context of IDS, having network traffic records may help to extract many valuable features. The following information will be collected: such as packet length, source and destination IP addresses, source, and destination port numbers, and the protocol used. etc. By selecting and using relevant features, data analysts can gain a more detailed understanding of the data and use this information to make informed decisions.

In machine learning, feature selection involves identifying a subset of features from the original dataset to use in further analysis or modeling. Feature selection can help reduce the

computational power required to build and run the model by using a smaller number of features, and it can also help improve the performance of the model by removing irrelevant, redundant, or noisy features. Using only the most relevant features can improve the model's accuracy and precision can be improved. In this project, we will use tools such as SelectKbest to select the most significant features and increase the accuracy of the classification models and to decrease computational overload.

4.4 Classification Model (Training Phase)

The effectiveness of machine learning algorithms can vary depending on the dataset they are applied to. In order to select the most suitable algorithm for a given model, it is necessary to analyze the performance of various machine learning algorithms. According to previous research [1][6][7], the support vector machine (SVM), random forest (RF), and decision tree (DT) algorithms are known to produce good results and have been chosen for use in the proposed system.

4.4.1 Support Vector Machine

SVM is an ML classifier that divides predictions into various categories. It involves the use of feature vectors, which are characteristics or features of the data, to make predictions about a particular event [28]. SVMs are a popular choice in machine learning due to their effectiveness in a variety of applications, such as intrusion detection, gene classification, etc. It is considered versatile algorithm for data analysis and prediction because it is capable of classification on both linear and non-linear data.

4.4.2 Random Forest

A random forest is a supervised learning algorithm that creates a model using multiple decision trees. It introduces randomness into the model by selecting a random subset of features to consider at each split, rather than selecting the most important feature. This results in a more diverse set of trees, which can improve the model's overall performance [29].

4.4.3 Decision Tree

Decision trees are a popular and effective method for supervised learning, particularly in classification tasks. They work by creating a tree-like model of decisions based on features of the data. At each decision point, or node, the tree splits into branches based on the value of a particular feature. The tree continues to split until it reaches a leaf node, which represents a prediction or

decision. One of the main advantages of decision trees is their interpretability. It is easy to understand how the model arrived at a particular decision by following the path through the tree. Decision trees can also handle both categorical and numerical data, and they are resistant to the impact of outliers [30].

5. Implementation

The system methodology flow that is shown in the figure below illustrates the steps that are involved in the implementation of the IDS.

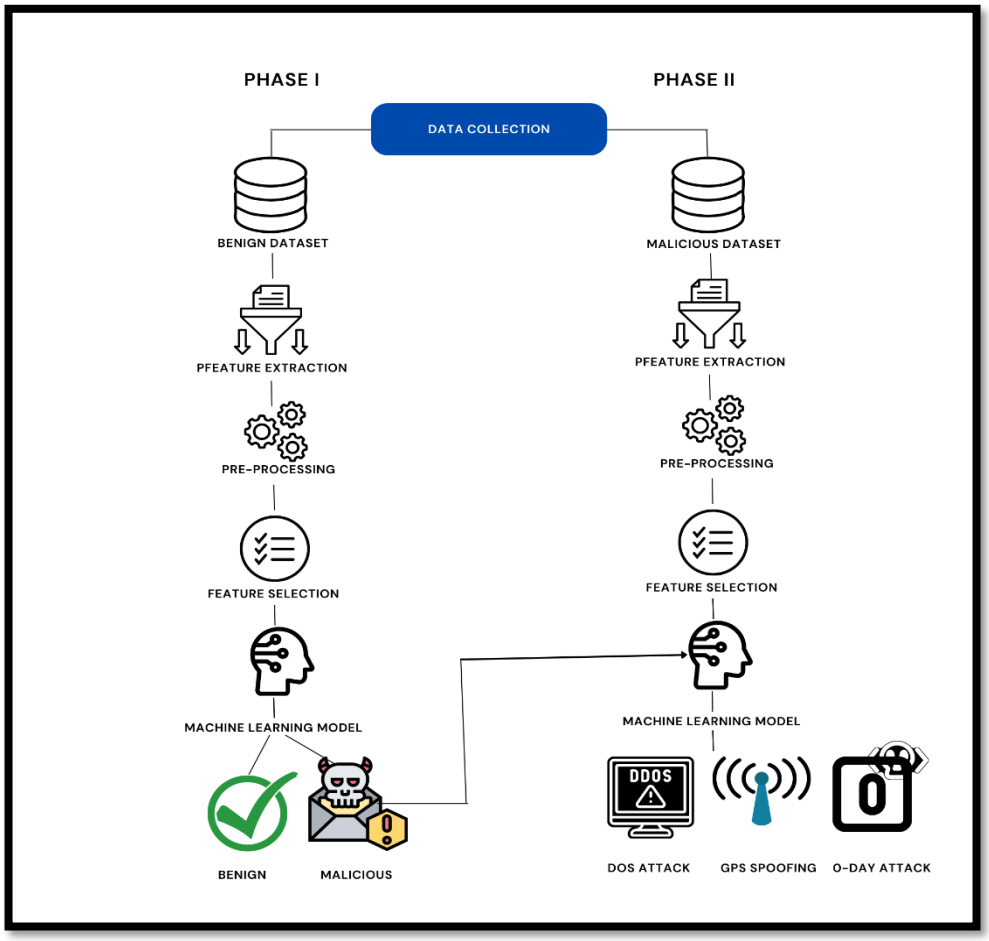

Figure 6: Two-Stage Classification

5.1 Dataset Collection and Cleaning

Data collection is the most significant step in training a machine learning model. The quality and diversity of the data used to train a machine learning model are crucial for its effectiveness and accuracy. Therefore, it is important to carefully gather and process a large and varied dataset during the data collection phase. Failing to do so can negatively impact the model's performance.

The proposed IDS requires a network-oriented dataset, as it works on the network level where almost all attacks on the drone are centralized. Since drones are vulnerable to variety of attacks, such as Denial-of-Service attack, hijacking control system, etc. it is essential that IDS has a holistic dataset that can accurately detect these attacks.

We have generated dataset for two phases of classification. The first stage includes One-Class classification where we generated dataset containing only normal UAV behaviour characteristics. Whereas in the second, we simulated various type of attacks on the drone to create a dataset that contains several attacks characteristics.

5.1.1 Dataset Collection Methodology

The data collection process was conducted in a controlled environment where we employed a drone, a laptop equipped with Wireshark (a packet capturer and analyzer), and a network adapter (Alpha) to monitor the communication between the drone and its controller.

We activated monitor mode on the network adapter and configured Wireshark to capture all traffic going to and from the drone. By doing so, we were able to record all the inbound and outbound traffic from the drone.

The following section explains the steps involved in data collection for two phases.

5.1.1.1 Data Collection for Stage I:

The stage I data collection process included the following steps:

- The controller was connected to the drone.
- The laptop was connected to the network adapter and the “Monitor Mode” was enabled on the network adapter.
- Wireshark was configured to capture all traffic going to and from the drone.
- The drone was flown around the test area and the data collection process was stopped after the drone had been flown around the test area for a sufficient amount of time.

5.1.1.2 Data Collection for Stage II:

The stage II data collection process included the following steps:

- The controller was connected to the drone.
- The laptop was connected to the network adapter and the "Monitor Mode" was enabled on the network adapter.
- Wireshark was configured to capture all traffic going to and from the drone.
- The drone was flown around the test area.
- Used another machine to simulate several attacks on the drone, mimicking real-world threats.
- Carefully captured and logged the attack traffic generated by the simulated attacks in Wireshark.
- The data collection process was stopped after the drone had been flown around the test area for enough time.

We ensured throughout the data collection process that the capturing process continued without interruption, preventing any potential data loss or corruption. We also took steps to ensure the quality of the data by eliminating any potential biases that could have affected the analysis.

5.1.2 Dataset pre-processing

Data pre-processing is performed to remove noisy, incomplete and inconsistent data and to ensure that the data is of high quality and relevance for analysis and modelling.

This stage is divided into the following sub-phases:

Cleaning:

We employed manual techniques to clean the data using Microsoft Excel.

- **Remove Duplicate Data**

  We removed the duplicate data by comparing each row with every other row in the dataset CSV file. To do this, we utilized the "Remove Duplicates" feature under the "Data" tab in Excel.

- **Remove Null Values**

  Null values can introduce inaccuracies in analysis; therefore, we located and eliminated the cells containing null values using Excel's "Find & Select" function. Under "Find &

Select," we navigated to the "Go To Special" option and selected "Blanks." All cells containing null values were highlighted and deleted.

- **Remove Empty Lines**

  Empty lines could disrupt the flow and readability of the dataset. Through careful selection and deletion, we removed these unnecessary gaps, presenting a clean and compact dataset that was easy to navigate and interpret.

- **Label Data**

  We identified the specific columns and manually entered the relevant labels into the corresponding cells.

Integration:

Combined data from multiple CSV files to create one dataset.

5.1.3 Dataset Collected

We collected a total of **6111** instances of the dataset. Out of these, **3775** instances were labeled as benign and **2336** were labeled as malicious. Within the malicious dataset, we further categorized the instances based on the types of attacks. Specifically, we had **1868** instances related to Denial of Service (DoS) attacks, **288** instances related to Hijack attacks, and **180** instances related to Man-in-the-Middle (MitM) attacks.

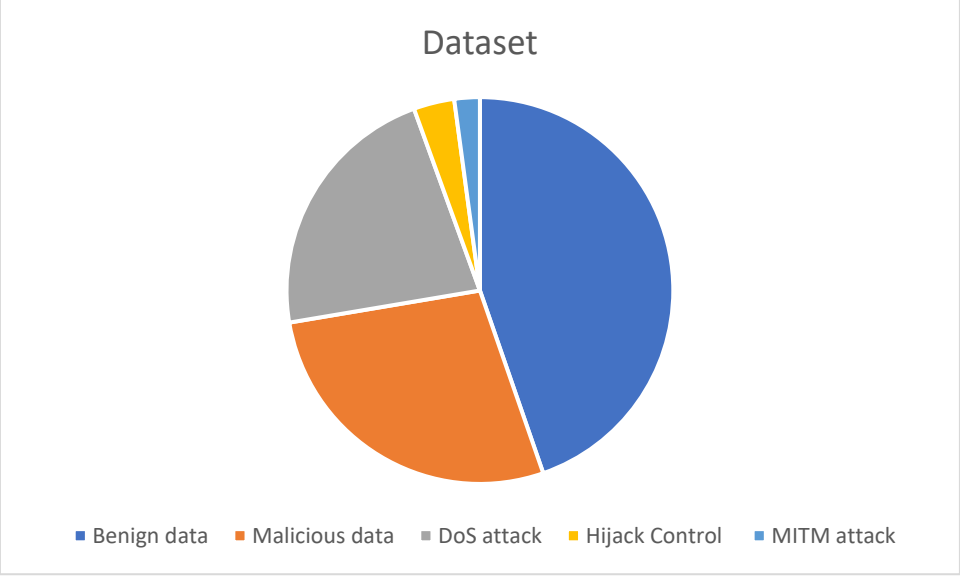

Figure 7: *Dataset Collected*

The data collected includes the commands issued by the controller to the drone, the port numbers on which the drone communicated, responses from the drone to the controller, telemetry data from the drone and much more.

5.2 Feature extraction

In this section, we will explore the process of feature extraction implemented in our project. Feature extraction is a crucial step in the developed machine learning-based Intrusion Detection Systems (IDS), as it involves transforming raw data into meaningful and representative features that is used then for detection and classification purposes. We will discuss the techniques and methods employed to extract relevant features from the drone traffic data and the extracted features resulted by implementing our methodology.

5.2.1 Feature Extraction Methodology

To delve into the process of feature extraction implemented in our project to analyze the captured traffic raw dataset. To extract relevant features from the traffic data, we utilized the NFStream library. NFStream proved to be a powerful and flexible tool that facilitates the extraction process and results in meaningful and representative features from network flows. The NFStream library, being a multi-platform Python framework, offers a fast, flexible, and intuitive approach to working with both online and offline network data. Its aim is to become a fundamental high-level building block for applicable network flow data analysis that aligned perfectly with our project objectives. NFStream takes as input raw network traffic pcap file or network interface name and extracts a wide range of flow-based features. The library offers various configuration options to specify flow characteristics and control the extraction process. For instance, we configured the following options [30]:

Configuration option	Description
Packet capture source	NFStream allows us to specify the source of the packet capture, which can be an interface name or an offline pcap file.
Idle timeout	Flows that remain idle (no packets received) for a duration exceeding the specified timeout value (in seconds) are considered expired and saved as a flow.
Active timeout	Flows that remain active for a duration exceeding the specified timeout value (in seconds) are considered expired and saved as a flow.

Statistical analysis	We enabled the post-mortem flow statistical analysis, which enhances the accuracy of our models. NFStream provides statistical analysis of flow data, extracting valuable insights for further analysis.
Split analysis	NFStream allows us to specify the sequence of first packet lengths for early statistical analysis. This feature provides additional granularity and aids in capturing important characteristics of the flows.
Max number of flows	We specified the maximum number of flows to capture before returning. This control allowed us to manage the flow data efficiently and avoid overwhelming the system.
Number of meters	NFStream provides the option to specify the number of parallel metering processes. This flexibility enabled us to optimize the performance of the feature extraction process.

Table 2: *Configuration Options*

We utilized these configuration options to tailor the behavior of the NFStream object to our project requirements. By leveraging the functionalities provided by NFStream, we successfully extracted numerous meaningful and representative features from the captured traffic. These features played a pivotal role in the creation of our dataset during the training phase and remained fixed throughout the deployment phase for real-time detection.

5.2.2 Extracted Features

Our proposed methodology for feature extraction yielded a comprehensive set of features for flows, encompassing both standard flow attributes and statistical analysis features. These extracted features served as essential inputs for our detection and classification models, enhancing their effectiveness and accuracy. The following 75 features are extracted:

expiration_id	src_port	dst_port	protocol	ip_version	vlan_id
tunnel_id	bidirectional_first_seen_ms	bidirectional_last_seen_ms	bidirectional_duration_ms	bidirectional_packets	bidirectional_bytes
src2dst_first_seen_ms	src2dst_last_seen_ms	src2dst_duration_ms	src2dst_packets	src2dst_bytes	dst2src_first_seen_ms
dst2src_last_seen_ms	dst2src duration_ms	dst2src packets	dst2src bytes	bidirectional min_ps	bidirectional_mean_ps
bidirectional_stddev_ps	bidirectional_max_ps	src2dst_min_ps	src2dst_mean_ps	src2dst_stddev_ps	src2dst_max_ps
dst2src_min_ps	dst2src_mean_ps	dst2src_stddev_ps	dst2src_max_ps	bidirectional_min_piat_ms	bidirectional_mean_piat_ms
bidirectional_stddev_piat_ms	bidirectional_max_piat_ms	src2dst_min_piat_ms	src2dst_mean_piat_ms	src2dst_stddev_piat_ms	src2dst_max_piat_ms
dst2src_min_piat_ms	dst2src_mean_piat_ms	dst2src_stddev_piat_ms	dst2src_max_piat_ms	bidirectional_syn_packets	bidirectional_cwr_packets
bidirectional_ece_packets	bidirectional_urg_packets	bidirectional_ack_packets	bidirectional_psh_packets	bidirectional_rst_packets	bidirectional_fin_packets
src2dst_syn_packets	src2dst_cwr_packets	src2dst_ece_packets	src2dst_urg_packets	src2dst_ack_packets	src2dst_psh_packets
src2dst_rst_packets	src2dst_fin_packets	dst2src_syn_packets	dst2src_cwr_packets	dst2src_ece_packets	dst2src_urg_packets
dst2src_ack_packets	dst2src_psh_packets	dst2src_rst_packets	dst2src_fin_packets	splt_direction	splt_ps
splt_piat_ms	application_is_guessed	application_confidence			

Figure 8: *Features Extracted*

All extracted features are defined and described by NFStream API documentation [30].

5.3 Feature Selection

Feature selection plays a crucial role in building an effective and efficient intrusion detection system (IDS) for Unmanned Aerial Vehicles (UAVs). In this research, we employed the SelectKBest method with the f_classif score function to select the most relevant features for both Stage 1 and Stage 2 of our proposed two-stage classification approach.

5.3.1 Stage I selected Features

For Stage 1, where we classify network traffic as either normal or potentially malicious, we applied feature selection on the available dataset using the (SelectKBest) method. We set (k) to 30, indicating our intention to select the top 30 features based on their relevance to the target variable.

The following features were selected for Stage 1 based on their relevance to classifying network traffic as normal or potentially malicious:

Feature	Description
bidirectional_psh_packets	Number of bidirectional PSH (Push) packets
bidirectional_rst_packets	Number of bidirectional RST (Reset) packets
bidirectional_fin_packets	Number of bidirectional FIN (Finish) packets
src2dst_syn_packets	Number of SYN (Synchronize) packets from source to destination
src2dst_cwr_packets	Number of CWR (Congestion Window Reduced) packets from source to destination
src2dst_ece_packets	Number of ECE (Explicit Congestion Notification Echo) packets from source to destination
src2dst_urg_packets	Number of URG (Urgent) packets from source to destination
src2dst_ack_packets	Number of ACK (Acknowledgment) packets from source to destination
src2dst_psh_packets	Number of PSH (Push) packets from source to destination
src2dst_rst_packets	Number of RST (Reset) packets from source to destination
src2dst_fin_packets	Number of FIN (Finish) packets from source to destination
dst2src_syn_packets	Number of SYN (Synchronize) packets from destination to source
dst2src_cwr_packets	Number of CWR (Congestion Window Reduced) packets from destination to source
dst2src_ece_packets	Number of ECE (Explicit Congestion Notification Echo) packets from destination to source
dst2src_urg_packets	Number of URG (Urgent) packets from destination to source
dst2src_ack_packets	Number of ACK (Acknowledgment) packets from destination to source
dst2src_psh_packets	Number of PSH (Push) packets from destination to source
dst2src_rst_packets	Number of RST (Reset) packets from destination to source
dst2src_fin_packets	Number of FIN (Finish) packets from destination to source
application_is_guessed	Indicates if the application or protocol is guessed
application_confidence	Confidence level in identifying the application or protocol
splt_direction_1	First flow packet direction (0: src2dst, 1: dst2src, -1: no packet)
splt_direction_2	Second flow packet direction (0: src2dst, 1: dst2src, -1: no packet)
splt_direction_3	Third flow packet direction (0: src2dst, 1: dst2src, -1: no packet)
splt_ps_1	Size of the first flow packet (depends on accounting_mode, -1 when there is no packet)
splt_ps_2	Size of the second flow packet (depends on accounting_mode, -1 when there is no packet)
splt_ps_3	Size of the third flow packet (depends on accounting_mode, -1 when there is no packet)
splt_piat_ms_1	Inter-arrival time (in milliseconds) between the first flow packet and the previous packet (always 0 for the first packet, -1 when there is no packet)
splt_piat_ms_2	Inter-arrival time (in milliseconds) between the second flow packet and the previous packet (always 0 for the first packet, -1 when there is no packet)

Figure 9: *Selected Features for Stage I*

These features provide valuable information about packet directions, sizes, and inter-arrival times, which can aid in identifying patterns and anomalies in the network traffic of UAVs.

5.3.2 Stage II selected Features

For Stage 2, the following features were selected to classify specific anomalies within potentially malicious traffic:

Feature	Description
expiration_id	Identifier related to the expiration/termination of a connection
src_port	Source port of the network traffic
dst_port	Destination port of the network traffic
protocol	Network protocol being used (e.g., TCP, UDP)
bidirectional_duration_ms	Duration of bidirectional communication in milliseconds
bidirectional_packets	Total number of packets exchanged in bidirectional communication
bidirectional_bytes	Total number of bytes exchanged in bidirectional communication
src2dst_duration_ms	Duration of communication from source to destination in milliseconds
src2dst_packets	Total number of packets exchanged from source to destination
src2dst_bytes	Total number of bytes exchanged from source to destination
dst2src_last_seen_ms	Time in milliseconds since the last communication from destination to source
bidirectional_stddev_ps	Standard deviation of packet sizes in bidirectional communication
bidirectional_max_ps	Maximum packet size observed in bidirectional communication
src2dst_stddev_ps	Standard deviation of packet sizes from source to destination
src2dst_max_ps	Maximum packet size observed from source to destination
bidirectional_stddev_piat_ms	Standard deviation of inter-arrival times between packets in bidirectional communication
src2dst_stddev_piat_ms	Standard deviation of inter-arrival times between packets from source to destination
application_confidence	Confidence level in identifying the application or protocol being used
splt_direction_2	Second flow packet direction (0: src2dst, 1: dst2src, -1: no packet)
splt_direction_3	Third flow packet direction (0: src2dst, 1: dst2src, -1: no packet)

***Figure 10**: Selected Features for stage II*

These features capture various aspects of network traffic behavior, including duration, packet counts, byte counts, statistical properties (such as standard deviation and maximum values), and application confidence. Additionally, splt_direction_2 and splt_direction_3 provide information about the second and third flow packet directions, respectively, following the same representation as described earlier.

The selected features will serve as crucial inputs for the subsequent stages of the intrusion detection system, aiding in accurately classifying anomalies and ensuring the security and dependability of UAV operations.

5.4 Building ML based model

5.4.1 Stage I Model building

The model is built using two datasets, one that contained benign data, and one that contained malicious data. This stage focused only on benign data to train the model. Benign data has been split into two parts, 80% for training and 20% for testing. Additionally, the malicious dataset was employed for testing in order to assess the model's performance in handling the malicious data.

To enhance the model's ability to identify patterns and make accurate predictions, feature selection methodologies were implemented. These methodologies aimed to select the most informative features, thereby augmenting the model's effectiveness.

Moreover, data normalization techniques were employed to eliminate biases that may have emerged due to varying measurement units. By applying these techniques, the model could effectively utilize the most pertinent features and guarantee uniform expansion throughout the dataset.

5.4.2 Stage II Model building

In this phase, the model is built using only malicious dataset. Similar to the first stage, the dataset was split two parts, 80% for training and 20% for testing. However, in this stage, only the malicious dataset was utilized for training and testing phase whereas in the first stage, only benign data was used for training but both benign and malicious datasets were used for the testing phase.

To enhance the model’s performance, feature selection techniques were employed by choosing the top ten features to maximize the model’s predictive capability. These features were then utilized in the training and testing processes across four different algorithms, specifically Random Forest, XGBoost, Decision Tree, and Naive Bayes.

Moreover, data normalization techniques were employed to eliminate biases that may have emerged due to varying measurement units.

5.4.3 Classification Models

The classifier can differentiate between various objects using specific features and machine learning algorithms. According to the related work, the following classifiers are chosen for further testing.

5.4.3.1 Local Outlier Factor

The Local Outlier Factor (LOF) algorithm has been used due to its proficiency in detecting anomalies that may exist within a given dataset. Our process began by creating an instance of the Local Outlier Factor (LOF) model with well-defined parameters. LOF is particularly effective in detecting outliers in datasets with varying densities or clusters. It has applications in various domains such as fraud detection, network intrusion detection, and outlier detection in sensor data. The algorithm is relatively easy to implement and offers a flexible approach to identifying anomalies in diverse datasets. Parameters were tested, listed and described in Table:

Parameter	Description	Values
Novelty	classifying new data as similar or different to the training set.	True
number of neighbors (k)	determines the size of the local neighborhood used to calculate the data point's density	5

Table 3: *Local Outlier Factor parameters*

5.4.3.2 One Class SVM

One-Class Support Vector Machines (One-Class SVMs) are a type of supervised machine learning algorithm used for anomaly detection. Unlike traditional SVMs that are trained with labeled data for classification, One-Class SVMs are trained with only one class of data points, typically representing the normal or non-anomalous instances. The algorithm learns to construct a decision boundary around the normal instances, aiming to encapsulate them within a high-dimensional feature space. Parameters were tested, listed and described in Table:

Parameter	Description	Values
kernel	Specifies the kernel type to be used in the algorithm.	rbf
nu	An upper bound on the fraction of margin errors and a lower bound of the fraction of support vectors. Should be in the interval (0, 1]	0.05

Table 4: *One Class SVM parameters*

5.4.3.3 XGBoost

XGBoost (eXtreme Gradient Boosting) is a powerful machine learning algorithm that belongs to the gradient boosting family. It is widely used for both regression and classification tasks due to its efficiency and excellent performance.

5.4.3.4 Naïve Bayes

Naive Bayes is a probabilistic machine learning algorithm based on Bayes' theorem and assumes independence among the features. It is commonly used for classification tasks, especially in text classification and spam filtering.

5.4.3.5 Decision Tree

A Decision Tree is a supervised machine learning algorithm that can be used for both classification and regression tasks. It is a non-parametric model that learns a hierarchical structure of if-else decision rules from the training data to make predictions.

The Decision Tree algorithm builds a tree-like structure where each internal node represents a decision based on a feature, and each leaf node represents a class label or a predicted value. The process of building the tree involves selecting the best feature at each node that provides the most significant information gain or reduction in impurity.

5.4.3.6 Random Forest Classifier

Random Forest Classifier is an ensemble machine learning algorithm that combines multiple decision trees to make predictions. It is primarily used for classification tasks and is known for its robustness and high accuracy.

The algorithm works by creating a "forest" of decision trees, where each tree is trained on a random subset of the training data and a random subset of features. During training, each tree

independently makes predictions, and the final prediction is determined by aggregating the votes or probabilities from all the trees.

5.4.4 Performance Evaluation for the Proposed Model

In accordance with our goals and model, we evaluated the performance of both the models by selecting the most appropriate performance metrics from the following:

5.4.4.1 Accuracy

Accuracy is a commonly used performance metric for classification models, including Random Forest Classifier. It measures the proportion of correctly classified instances out of the total number of instances in a dataset. It is calculated by dividing the number of correct predictions by the total number of predictions and multiplying by 100 to express it as a percentage. It is determined as follows:

$$Accuracy = \frac{Number\ of\ correct\ Predictions}{Total\ number\ of\ Prediction} \times\ 100$$

5.4.4.2 Precision

It is the proportion of true positive findings to total true and false positive findings. It works best when the goal is focused on true positives and false positives rather than false negatives.

Precision may be expressed as:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

5.4.4.4 F1-Score

The F1 score is affected by precision and recall and is greatest when both equal to one. It is a useful metric to consider when evaluating the performance of a classification model. It is particularly useful when it is important to achieve a balance between precision and recall, or when the cost of false positives and false negatives are not equal.

It may be computed mathematically as follows:

$$F1 = 2 \times \frac{Prediction \times Recall}{Prediction + Recall}$$

5.4.4.5 Confusion Matrix

A confusion matrix is used to determine the parameters that reflect the performance of the model i.e., precision, accuracy, and recall.

The confusion matrix is critical in deciding two outcomes in a two-class, or binary classification problem. In ML, these variables represent numerical values, and the outputs might be positive or negative. The confusion matrix uses rate as a measure factor. It has four types:

- True Positive Rate: The percentage of times a classifier accurately predicts intended results.

$$True\ Positive\ Rate\ (TPR) = \frac{True\ Positive}{Positive}$$

- True Negative Rate: The frequency with which a classifier accurately predicts unfavorable events.

$$True\ Negative\ Rate\ (TNR) = \frac{True\ Negative}{Negative}$$

- False positive Rate: Represents how frequently a classifier is inaccurate in predicting desired results.

$$False\ Positive\ Rate\ (FPR) = \frac{False\ Postive}{Negative}$$

- False negative Rate: Corresponds to the number of times a classifier predicts unfavorable outcomes inaccurately.

$$False\ Negative\ Rate\ (FNR) = \frac{False\ Negative}{Positive}$$

In order to achieve optimal performance, it is important to have a high TPR, TNR, and a low FNR, FPR.

5.5 Deployment

After building the main models of our project, the deployment phase started, which involves the implementation of a real-time two-stage Machine Learning (ML) based Intrusion Detection System (IDS) for detecting and classifying malicious drone traffic. The deployment phase concentrates on the practical aspects of integrating the developed models into a real-time IDS. Once the two-stage classification models were trained and tested, we exported them as pkl files using the pickle module. This format allows for easy storage and retrieval of trained models. To build the real-time IDS, we developed a customized Python script, leveraging the capabilities of nsfstream library. NFStream provides utilities for capturing and analyzing network traffic in real-time. We operated the following key utilities to implement our IDS effectively:

1. BPF Filter: The BPF (Berkeley Packet Filter) filter was utilized to selectively capture the traffic and filter only the drone-related incoming and outgoing network traffic. This filter acted as a pre-processing step to focus our analysis on the relevant data.
2. Promiscuous Mode: NFStream provides a promiscuous mode, which allows real-time traffic analysis from a specific monitoring port. This mode was vital in capturing and monitoring drone network traffic, ensuring accurate detection and classification.
3. NFPlugins: NFPlugins is a powerful feature of the NFSstream library that allows us to extend its functionality and incorporate custom analysis functions. In our project, we leveraged NFPlugins to import and analyze real-time traffic, enabling us to apply our trained models for detection and classification.

By combining the aforementioned utilities with our customized Python script, we successfully built a real-time IDS for detecting and classifying malicious drone traffic. The script continuously captured and analyzed network traffic, filtering out irrelevant data using the BPF filter. The captured traffic was then passed through our anomaly detection model to identify abnormal patterns. Afterward, the classified traffic was fed into the traffic classification model to assign appropriate labels to malicious traffic potential type.

6. Results

Result of the algorithms for stage I:

Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-score**
Local Outlier Factor (LoF)	0.82	0.75	0.82	0.77
One-Class SVM (OCSVM)	0.83	0.72	0.87	0.79

Table 5: *Classification Metrics for Stage I*

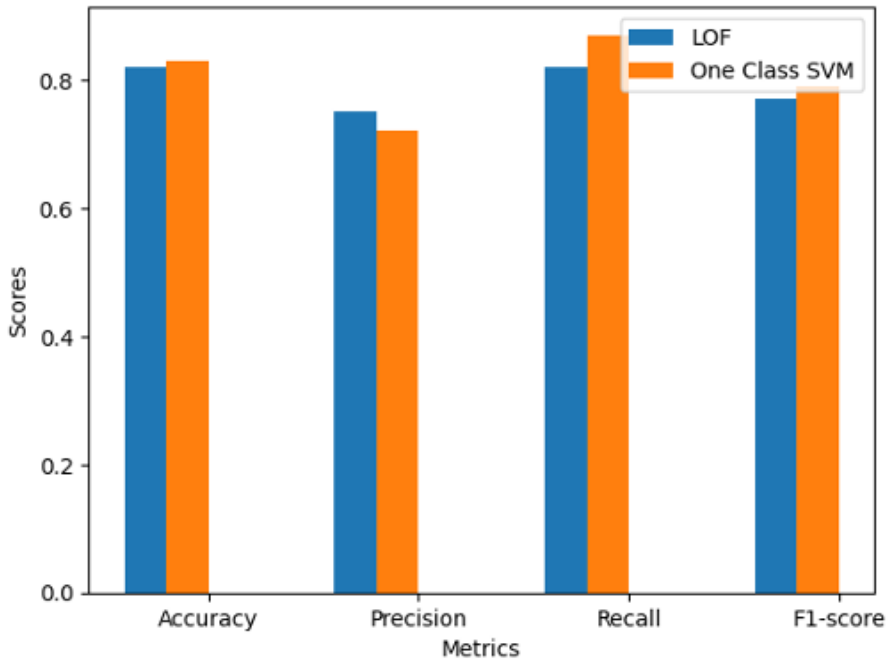

Figure 11: *Result of algorithms for stage I*

When it comes to accuracy, LoF and OCSVM perform similarly. However, OCSVM has slightly higher accuracy, recall and f1-scores than LoF. This suggests that OCSVM is better at identifying benign from malicious while also avoiding false positives.

Result of the algorithms for stage II:

Algorithm	Accuracy	Precision	Recall	F1-score
Random Forest	0.97	0.95	0.88	0.91
XGBoost	0.97	0.96	0.88	0.91
Naïve Bayes	0.66	0.38	0.39	0.35
Decision Tree	0.96	0.91	0.89	0.90

Table 6: *Classification Metrics for Stage II*

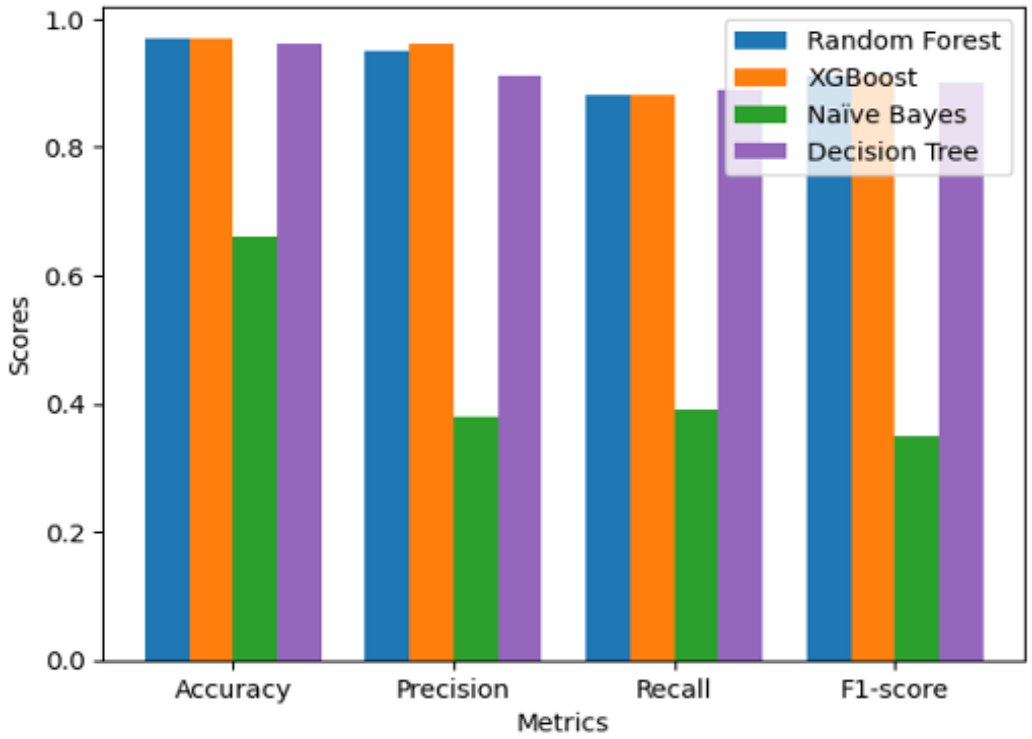

Figure 12: *Result of algorithms for stage II*

Random Forest and XGBoost achieved similar high accuracy of 0.97, indicating their ability to correctly classify instances. These algorithms also demonstrated comparable precision, recall, and F1-scores, with values ranging from 0.88 to 0.91.

In contrast, Naïve Bayes had a significantly lower performance with an accuracy of 0.66 and lower precision, recall, and F1-scores ranging from 0.35 to 0.39. Decision Tree performed well overall, with an accuracy of 0.96 and precision, recall, and F1-scores ranging from 0.89 to 0.91.

Overall, OCSVM is the best algorithm among the two for one-class classification. It had better performance with feature selection which is crucial for accurate classification and minimizing false positives. Random Forest is the best choice among the four algorithms for multi-class classification. Its robustness to noise, ensemble learning approach, feature importance estimation, and efficiency with high-dimensional data make it a suitable choice for our IDS.

7. CONCLUSION

In conclusion, the two-stage classification approach presented in this report offers a promising solution for detecting and preventing security breaches in UAVs. It effectively addresses the challenge of identifying and classifying all types of attacks, including zero-day attacks.

During the implementation phase, we followed a structured process which involved collecting and processing data, feature extraction, feature selection, training the models, and evaluating their performance using appropriate metrics. The use of One-Class SVM (OCSVM) in stage I and Random Forest (RF) algorithm in stage II for model training proved to be exceptionally well-suited for this system.

Overall, by leveraging the strengths of OCSVM and RF within the two-stage classification approach, our system achieves a comprehensive and effective detection of intrusions in UAVs.

To enhance the present contribution and suggest avenues for future research, it is recommended to apply this innovative methodology to diverse UAVs manufactured by other companies, utilizing more extensive and diverse datasets encompassing various types of attacks. This undertaking aims to augment accuracy and mitigate instances of false positive outcomes.

8. REFERENCES

[1] "Map of World Wide Drone Incidents - Dedrone." [Online]. Available: https://www.dedrone.com/resources/incidents/all

[2] "Commercial Drone Market Size." [Online].Available: https://www.grandviewresearch.com/industry-analysis/ global-commercial-drones-market

[3] F. Aviation Administration, "FAA National Forecast FY 2019-2039 Full Forecast Document and Tables," Tech. Rep.

[4] ZHANG, R. (2022). Intrusion detection system in a fleet of drones. Université Fédérale. Retrieved from https://www.theses.fr/2022ESAE0003.pdf.

[5] Khoei, T. T., Gasimova, A., Ahajjam, M. A., Shamaileh, K. A., Devabhaktuni, V., & Kaabouch, N. (2022). A comparative analysis of supervised and unsupervised models for detecting GPS spoofing attack on uavs. 2022 IEEE International Conference on Electro Information Technology (EIT). https://doi.org/10.1109/eit53891.2022.9813826

[6] Detecting signal spoofing attack in uavs using machine learning models. (n.d.). Retrieved December 2, 2022, from https://www.researchgate.net/publication/352467370_Detecting_Signal_Spoofing_Attack_in_UAVs_Using_Machine_Learning_Models

[7] Perera, P., & Patel, V. M. (n.d.). Learning deep features for one-class classification. IEEE.

[8] Park, K. H., Park, E., & Kim, H. K. (2020). Unsupervised Intrusion Detection System for Unmanned Aerial Vehicle with Less Labeling Effort. arXiv. https://doi.org/10.48550/arXiv.2011.00540

[9] Cyted. (n.d.). Retrieved December 6, 2022, from https://cyted.org/sites/default/files/network-based_intrusion_detection_a_one-class_classification_approach.pdf

[10] The UNSW-NB15 Dataset: UNSW Research. The UNSW-NB15 Dataset | UNSW Research. (n.d.). Retrieved December 17, 2022, from https://research.unsw.edu.au/projects/unsw-nb15-dataset

[11] Aldaej, A.; Ahanger, T.A.;Atiquzzaman, M.; Ullah, I.;Yousufudin, M. Smart Cybersecurity Framework for IoT-Empowered Drones: Machine Learning Perspective. Sensors 2022, 22, 2630. https://doi.org/10.3390/s22072630

[12] Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE symposium on computational intelligence for security and defense applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.

[13] Siddique, K.; Akhtar, Z.; Khan, F.A.; Kim, Y. KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research. Computer 2019, 52, 41–51.

[14] Park, J., Kim, Y., & Seok, J. (2022). Prediction of Information Propagation in a Drone Network by using Machine Learning. The School of Electrical Engineering, Korea University.

[15] Zhao, L. (n.d.). Unmanned aerial vehicles (UAVs) in civil applications. Retrieved from https://mason.gmu.edu/~lzhao9/materials/data/UAV/

[16] Pawlak, J. et al. (2021) "A machine learning approach for detecting and classifying jamming attacks against OFDM-based uavs," Proceedings of the 3rd ACM Workshop on Wireless Security and Machine Learning [Preprint] Retrieved December 17, 2022, from https://doi.org/10.1145/3468218.3469049

[17] Anderson, J.P.(1980) "Computer security threat monitoring and surveillance": http://csrc.nist.gov/publications/history/#ande80

[18] Singh, H. (2022, May 13). Host-based Intrusion Detection System – Overview and HIDS vs NIDS. Cyphere. Retrieved December 12, 2022, from https://thecyphere.com/blog/host-based-ids/

[19] Mhawi, D.N.; Aldallal, A.; Hassan, S. Advanced Feature-Selection-Based Hybrid Ensemble Learning Algorithms for Network Intrusion Detection Systems. Symmetry 2022, 14, 1461. https://doi.org/10.3390/ sym14071461

[20] Foote, K. D. (2022, January 20). A brief history of machine learning. DATAVERSITY. Retrieved December 17, 2022, from https://www.dataversity.net/a-brief-history-of-machine-learning

[21] KDD Cup 1999 data - University of California, Irvine. (n.d.). Retrieved December 18, 2022, from https://kdd.ics.uci.edu/databases/kddcup99/kddcup99

[22] Drone forensics: VTO Labs. vtolabs. (n.d.). Retrieved December 18, 2022, from https://www.vtolabs.com/drone-forensics

[23] Unmanned Aerial Vehicle (UAV) intrusion detection datasets. Liang Zhao's Homepage. (n.d.). Retrieved December 18, 2022, from http://mason.gmu.edu/~lzhao9/materials/data/UAV/

[24] KDD Cup 1999 Data. (n.d.). Retrieved December 17, 2022, from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[25] Simulation. PaparazziUAV. (n.d.). Retrieved December 17, 2022, from https://wiki.paparazziuav.org/wiki/Simulation

[26] Zhao, L., Zeng, K., & others. (n.d.). Anomaly detection in large-scale networks via graph convolutional recurrent autoencoder. Retrieved from https://people-ece.vse.gmu.edu/~kzeng2/publications/2018/R1450p-zhaoA-KDD18.pdf

[27] Support Vector Machine (SVM) algorithm - javatpoint. www.javatpoint.com. (n.d.). Retrieved December 2, 2022, from https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

[28] Random Forest classifier: A complete guide to how it works in Machine Learning. Built In. (n.d.). Retrieved December 2, 2022, from https://builtin.com/data-science/random-forest-algorithm

[29] Machine learning decision tree classification algorithm - javatpoint. www.javatpoint.com. (n.d.). Retrieved December 2, 2022, from https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

[30] APIs documentation. nfstream. (n.d.). https://www.nfstream.org/docs/api