Cyber-Physical Intrusion Detection System for Unmanned Aerial Vehicles

Samuel Chase Hassler, Umair Ahmad Mughal, and Muhammad Ismail, *Senior Member, IEEE*

***Abstract*—The increasing reliance on unmanned aerial vehicles (UAVs) has escalated the associated cyber risks. While machine learning has enabled intrusion detection systems (IDSs), current IDSs do not incorporate cyber-physical UAV features, which limits their detection performance. Additionally, the lack of public UAV's cyber and physical datasets to develop IDS hinders further research. Therefore, this paper proposes a novel IDS fusing UAV cyber and physical features to improve detection capabilities. First, we developed a testbed that includes UAV, controller, and data collection tools to execute cyber-attacks and gather cyber and physical data under normal and attack conditions. We made this dataset publicly available. The dataset covers a range of cyber-attacks including denial-of-service, replay, evil twin, and false data injection attacks. Then, machine learning-based IDSs fusing cyber and physical features were trained to detect cyber-attacks using support vector machines, feedforward neural networks, recurrent neural networks with long short-term memory cells, and convolutional neural networks. Extensive experiments were conducted on varying complexity and range of attack training data to explore whether (a) fusion of cyber and physical features enhances detection performance compared to cyber or physical features alone, (b) fusion enhances detection when IDS is trained on a single attack type and tested on unseen attacks of varying complexity, (c) fusion enhances performance when the range of attack training data increases and models are tested on unseen attacks. Answering these research questions provides insights into IDS capabilities using cyber, physical, and cyber-physical features under different conditions.**

***Index Terms*—UAVs, cyber-physical systems, intrusion detection systems, and machine learning.**

I. Introduction

UNMANNED aerial vehicles (UAVs) are going through rapid adoption in many sectors from traffic analysis to smart agriculture to military operations. As UAVs are performing different tasks, they are often equipped with various sensors, actuators, and processing units to aid their particular task. These UAVs are cyber-physical systems that collect measurements using sensors, receive command signals from ground controllers, do some processing, and then take some physical actions using actuators. Like all cyber-physical systems, UAVs suffer from several security vulnerabilities that hinder their adoption [1]. It is not hard to imagine the potential damage if, for example, adversaries target common surveillance UAVs causing them to malfunction. Likewise, a malicious actor could potentially hijack a fleet of UAVs and reroute them to crash into urban areas. Apart from the obvious possible physical damages, data theft is also a great threat. Since nearly all modern drones come equipped with at least one camera, hijacked video feeds could give competitors access to private corporate information. All of these examples show the importance of securing UAVs against cyber-attacks.

Research has been conducted to develop intrusion detection systems (IDSs) that ensure the secure operation of UAVs against cyber-attacks. The developed IDSs aim at detecting replay attacks [2], hijacking attempts [3], spoofing attacks [4], false data injection (FDI) attacks [5], and denial-of-service (DoS) attacks [6]. In addition, systems have been developed to detect faults during the UAV operation [7]. However, existing IDSs have a common limitation as they do not treat UAVs as cyber-physical systems. Instead, current IDSs rely solely on either physical features of the UAV (e.g., roll, pitch, yaw angles, speed, temperature, etc.) or cyber features (e.g., IP addresses, MAC addresses, port numbers, etc.). Yet, UAVs are cyber-physical systems equipped with sensors, radios, and actuators that integrate computational and physical components. Therefore, strategies relying solely on cyber or physical features do not provide a complete representation of the UAV system, resulting in limited detection performance. Moreover, the existing works have not investigated the impacts of cyber and physical fusion on IDS capabilities under varying conditions of complexity and range of the attack training data. Attack complexity refers to the sophistication of the attack that was used to train the IDS and how this impacts the ability of the IDS to detect unseen attacks in the testing stage. On the other hand, the range of the attack training data relates to the different types of attacks used for model training and how this impacts the IDS ability to detect unseen attacks. Therefore, this paper investigates the following research questions:

- Will the fusion of cyber and physical features improve detection compared to cyber or physical features alone?
- Will the complexity level of the attack training data impact the ability of the IDS to detect unseen attacks with various complexities?
- Will the range of the attacks included in the training data impact the ability of the IDS to detect unseen attacks with various complexities?

Answering the aforementioned research questions faces several key challenges. First, developing effective IDSs requires datasets that capture system behavior under both normal and attack conditions across cyber and physical domains. However, existing UAV datasets do not provide both cyber and physical features, highlighting the need to develop an experimental testbed and data collection methodology. Second,

S. C. Hassler, U. A. Mughal, and M. Ismail are with the Department of Computer Science, College of Engineering, Tennessee Tech University, Cookeville, TN, 38501, USA email: {schassler42, uamughal42, mismail}@tntech.edu

This work is supported by NSF EPCN Award 2220346.

equipping IDS with numerous cyber and physical features may overwhelm the model, degrading detection performance. Therefore, a methodology is needed to identify the most salient cross-domain features that boost detection capabilities. Finally, balancing model computational complexity and performance requires exploring a diverse set of strategies, optimizing their architectures, and benchmarking their detection performance.

To address these challenges, we carried out the following:

- We developed a testbed that consists of UAV, controller, and data collection tools. Using this testbed, we executed four attack types on the UAV, namely, de-authentication DoS, replay, evil twin, and FDI attacks. We collected and published in [8] cyber-physical datasets that represent UAV operations under normal and attack conditions.
- We performed Shapley additive explanations (SHAP) analysis to identify the most effective cyber and physical features for each model. Also, we carried out a grid search analysis to identify the optimal structure for each machine learning model. Finally, we studied the learning curves of each model to examine that a sufficient amount of data has been provided to each model and that each model is not under-fitting or over-fitting.
- We used the datasets to train and test a set of IDSs based on support vector machine (SVM), feedforward neural network (FNN), recurrent neural network with long-short-term-memory units (LSTM-RNN), and 1D convolutional neural network (CNN). We compared three variants for each model, namely, training and testing using cyber-only, physical-only, and fused cyber-physical features.
- To provide answers to the aforementioned research questions, we conducted extensive experiments involving two assessment scenarios, namely, the complexity of attack training data and the range of attack training data, based on the developed models (SVM, FNN, LSTM, and 1D-CNN) and their variants (cyber, physical, and cyber-physical). The first scenario reflects cases when models are trained on a single type of attack and tested against other unseen attacks. The second scenario reflects cases when the range of attacks considered in the training datasets increases and then the model is tested against other unseen attacks.

The rest of this paper is as follows. Section II reviews the related works and highlights their limitations. Section III discusses the testbed and the methodology to collect the benign and malicious datasets. Section IV presents the machine learning models and hyper-parameter optimization. Section V presents the detection and SHAP analysis results and provides discussions. Finally, conclusions are given in Section VI.

II. Related Works

This section reviews the published works in the area of IDS in UAVs. To the best of the author's knowledge, there is no research work that explores the fusion of cyber-physical data for intrusion detection in UAVs. Existing works either use cyber or physical data to develop the IDS. Furthermore, the vast majority of existing research uses pre-existing cyber datasets that were not taken from actual UAV communications. In the following, we divide the related works into two categories, namely, IDS based on cyber features and IDS based on physical features. The limitations of these categories highlight the contributions of our work.

A. IDS based on Cyber Features

Most of the existing research focuses on developing IDSs of UAVs using cyber features. For instance, Sedjelmaci et al. addressed in [5] the balance between detection rates and false positive rates through the use of a Bayesian game-theoretic methodology. The dataset used in [5] is obtained via simulation and is not publicly available. Zhang et al. developed in [6] a hybrid IDS capable of monitoring a fleet of UAVs by analyzing Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) network traffic sent between the ground controller and the UAV. The work in [6] used wavelet leader multi-fractal (WLM) analysis along with robust estimation to form the basis of the IDS. Ferrag and Maglaras developed in [7] a framework for delivery UAVs using both blockchain and an IDS called DeliveryCoin. The IDS in [7] was developed by testing a variety of machine learning models, namely, decision tree, SVM, CNN, and recurrent neural network, and finding the best model that is capable of not only detecting attacks but also classifying them. Tan et al. proposed in [9] an IDS for UAVs using a deep belief network based on stacked restricted Boltzmann machines. The IDS model in [9] was optimized using particle swarm optimization and the results were found to outperform a neural network with three layers, SVM, AdaBoost, and a deep neural network with eight layers. Ouiazzane et al. developed in [10] an IDS model for UAVs using various agents (sniffers, filters, feature selectors, decision-makers, etc.). Using a decision tree structure, the IDS in [10] was able to correctly classify all instances in the CICIDS-2017 intrusion detection evaluation dataset. Shrestha et al. developed in [11] an IDS for UAVs that runs on 5G communications. In the testing of [11], seven different models were considered, including linear regression, linear discriminant analysis, K-nearest neighbor, decision tree, and Gaussian Naive Bayes. From this subset of models, it was found that decision trees resulted in the best overall performance across various attacks. Recently, Praveena et al. employed in [12] a reinforcement learning model optimized through black widow optimization on the NSL-KDD dataset. It was shown in [12] that the proposed methodology outperforms all the existing models including improved deep belief network, selective and inter-related distillation, deep learning, density peak clustering-deep belief network, adapted K-nearest neighbors, decision tree, AdaBoost, random forest, and SVM. Bouhamaed et al. [13] adopted deep Q-learning for the detection of cyber-attacks. The authors used the CICIDS-2017 dataset for the evaluation of the proposed model. Kou et al. [14] proposed a model consisting of a deep autoencoder and a CNN for anomaly detection in the UAV communication of a software-defined network (SDN). The authors evaluated the performance of the proposed model using the InSDN dataset. Han et al. [15] proposed IDS to secure the packet information using the hierarchical LSTM model. The proposed model was evaluated using the CICIDS-2017 dataset. Mehmoodat et

TABLE I
SUMMARY OF RELATED WORKS

Ref.	Year	Research Focus	Detection Mechanism	Dataset
[5]	2016	UAV-Aided Networks	Bayesian Game-Theory	Simulated Cyber
[6]	2018	UAV Communication	Signature-based	Simulated Cyber
[7]	2019	UAV-Delivery	SVM & KNN	Irrelevant Cyber (CSE-CIC-IDS-2018)
[9]	2019	UAV Communication	Deep Belief Network	Irrelevant Cyber (KDD-Cup-99)
[10]	2020	Ad-hoc Communication	Decision Tree	Irrelevant Cyber (CICIDS-2017)
[11]	2021	Cellular Connected UAV Networks	Decision Tree & KNN	Irrelevant Cyber (CSE-CIC IDS-2018)
[12]	2022	UAV Networks	Reinforcement Learning	Irrelevant Cyber (NSL-KDD)
[13]	2021	UAV Coverage	Deep Q-learning	Irrelevant Cyber (CICIDS-2017)
[14]	2022	UAV in SDN	Autoencoder & CNN	Irrelevant Cyber (InSDN)
[15]	2023	UAV Network	Hierarchical LSTM	Irrelevant Cyber (CICIDS-2017)
[16]	2022	UAV Communication	SVM & Random Forest	Simulated Cyber
[17]	2021	UAV Communication	Autoencoders	Actual Physical
[18]	2021	GNSS Communication	Parameters-based	Actual Physical
[19]	2022	GNSS Communication	Kullback–Leibler Divergence	Actual Physical
[20]	2019	UAV Monitoring	1D-CNN	Actual Physical
[21]	2022	UAV Swarm	Deep Neural Network	Simulated Physical

al. [16] simulated the UAV communication environment in a COOJA simulator and generated the simulated traffic data for the IDS. The authors used the random forest, SVM, and K-nearest neighbor for detection purposes.

A common factor in all of the aforementioned works is that none of the IDSs was developed and tested using real communications taken from an actual UAV. Instead, all of the IDS models have been developed and tested using cyber data that is either simulated, e.g., [5], [6], [16], or adopted from public intrusion detection datasets irrelevant to UAVs, e.g., [7], [9]–[11], [13]–[15].

B. IDS based on Physical Features

The vast majority of research carried out in this area focuses on the cyber aspects, as existing works claim that research can still be conducted through the use of public cyber datasets irrelevant to UAV communications. However, there are a few examples that found ways to incorporate physical/behavioral features of UAVs. For instance, Park et al. used in [17] two UAV-specific datasets: (a) the UAV dataset that contains simulated flight logs under normal conditions and cyber-attack scenarios, and (b) the AirLab failure and anomaly dataset that consists of actual drone flight logs where component failures occurred. Using a set of extracted and engineered features from the physical data, autoencoders were trained to detect anomalies. Basan et al. developed in [18] a lightweight methodology for detecting global navigation satellite system (GNSS) spoofing attacks on UAVs via monitoring a few parameters and implementing a Poisson distribution model to find rare events. To validate the proposed model, the work in [18] recorded GNSS spoofing attacks on UAVs and successfully detected each instance. The following year, Basan et al. improved in [19] upon [18] using a Kullback-Leibler divergence model that is capable of detecting GNSS spoofing attacks without requiring a large sample of benign data. Ahn et al. [20] proposed a machine learning-based mechanism to detect anomalies in UAV behavior. The authors used 1D-CNN to identify the abnormal behavior of UAVs. In [20], the authors used only the physical features of the UAV for IDS training and evaluation purposes. Khanapuri et al. [21] used a deep neural network to design IDS for a swarm of UAVs working in a cooperative localization. However, the authors in [21] used simulated physical features for the IDS training and evaluation.

C. Limitations

Table I highlights the key features in the related works. As aforementioned, there is a great need for a dataset that monitors physical logs as well as cyber communications in UAVs during normal operations as well as under a cyber-attack. One of the contributions of our paper is the development and publishing of such a cyber-physical dataset [8] to promote further research in this area using datasets collected from an actual UAV. Moreover, all the aforementioned works relied on either cyber or physical datasets and never published correlated cyber and physical UAV data. Furthermore, our paper also presents findings for testing the hypothesis of fusing physical and cyber data to develop a more effective IDS under different scenarios involving various attack complexities and ranges.

III. DATASET GENERATION

Since we target a machine learning-based IDS, dataset availability is a major concern. In our case, there is no publicly available dataset that includes correlated physical and cyber data monitoring a UAV in flight under normal operation and cyber-attack conditions. Therefore, one of our contributions is the creation and publication of a dataset in [8] that meets such goals. This section details the process of creating the required cyber-physical dataset.

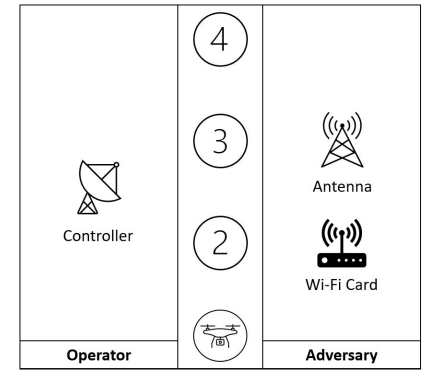
A. Equipment

The physical equipment used to create the dataset used in this paper includes DJI Tello EDU drone and accessories [22], ALFA AWUS036ACH antenna [23], computer-1 used as a controller to fly the UAV over pre-specified routes, and computer-2 with WiFi card operating in the monitor mode. The setup of the testbed is shown in Fig. 1.

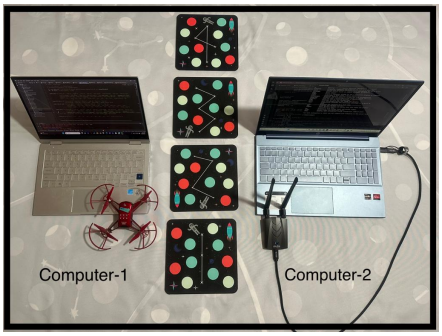
The DJI Tello EDU is a lightweight (80 g) programmable drone with dimensions of $98 \times 92.5 \times 41$ mm. It can be

programmed using the Tello SDK with Python, Scratch, or Swift. The drone provides approximately 13 minutes of flight time. It is equipped with a front-facing 720p camera for video transmission. It uses a vision positioning system that includes a front camera and two 3D infrared sensors, which are equipped at the bottom of the drone, for stable hovering. The drone also contains a time-of-flight (ToF) distance sensor, IMU, and barometer. It communicates over 2.4GHz 802.11n WiFi using the UDP protocol. The programmability of the Tello EDU via the SDK, its sensor suite, and its open software platform make it an ideal UAV for research and development.

Computer-1 was used to emulate a controller or ground control station used in operations, which will be discussed in the next subsection. This computer contains a number of Python scripts developed to communicate with the UAV and collect physical readings in real time along the flight. This computer can be thought of as the legitimate operator of the UAV. The ALFA AWUS036ACH antenna was used as a point to launch attacks while still recording network traffic through a WiFi card. Computer-2 was running a Linux distribution called ParrotOS (Security Edition) [24] that comes equipped with a large number of penetration testing tools that were helpful in the creation of the cyber part of this dataset. The main software used on this computer were Aircrack-ng [25], Tcpdump [26], and Wireshark [27]. This computer can be thought of as the adversary, which used the Alpha AWUS036ACH antenna and the computer's internal WiFi card to perform attacks and collect cyber data.



(a)



(b)

Fig. 1. Illustration of the testbed. (a) Sketch of the layout, (b) Photo of the testbed. In the middle, there are the 4 mission pads that the Tello drone moves in between. The left side shows the operator along with the controller (computer-1), and the right side shows the adversary and their equipment (computer-2 and antenna)

B. Data Collection Methodology

The dataset was gathered in two stages, namely, (a) flights under normal conditions without any cyber-attacks and (b) the same flights under four types of cyber-attacks. The first stage begins with computer-2 putting their WiFi card into wireless mode and running a tool called airodump-ng from Aircrack-ng to capture raw IEEE 802.11 frames. This tool listens to all the WiFi traffic across all channels, finds access points, and lists their basic service set identifier (BSSID). For each BSSID, various attributes are collected such as the channel, authentication protocol, encryption algorithms, cipher, and the extended service set identification (ESSID). If a BSSID is already known, there are filter options to display only traffic associated with that particular access point. For this stage of data collection, airodump-ng is used to collect all the network traffic (cyber data) sent to and from the Tello drone.

From here, the Tello drone is powered on and creates its own wireless access point for clients to connect (through computer-1) and send commands. Then, computer-1 runs a Python script that connects to the UAV and begins sending commands to control the UAV along a predefined path, as shown in Fig. 1. The Tello comes with four mission pads that were placed in a line approximately 76 cm from each other. The UAV takes off from a mission pad labeled as 1 and proceeds to mission pad 2, 3, and 4 in succession while stopping above each pad. Once the UAV gets to the 4th pad, it rotates 180 degrees and repeats the process in reverse order back to mission pad 1. When the UAV is above each pad, there was a 10% chance to perform a random action, such as moving back and forth, moving up and down, doing a flip to the left or right, etc. This flight path was designed to emulate a route used in smart farming, e.g., checking a row of plants for disease or spraying fertilizer. Other applications may also involve surveillance. Throughout the entire process, the UAV is sending the status of various physical measurements back to computer-1 every 0.5 seconds. These will make up the physical features in the dataset, which include measurements such as roll, pitch, yaw, speed along each axis, temperature of motors, etc. Once the UAV has completed the route and returned to mission pad 1, it rotates 180 degrees again and lands, ready to start over again. From here, the UAV is powered off, and airodump-ng running on computer-2 stops and the collected data is saved into a PCAP file which will become part of the cyber dataset.

The aforementioned entire process makes up one flight, and a total of 35 flights were performed. The first 20 flights have a standard 10% chance of a random action, the next 10 flights have the rate increased to 20%, and the final 5 flights with 50% chance of a random action from the UAV.

C. Cyber-attacks Under Consideration

The second stage follows the same process as the first stage but with a cyber-attack launched at a random point along the UAV's route. Four attacks were selected to be launched against the UAV, namely, de-authentication, replay, evil twin,

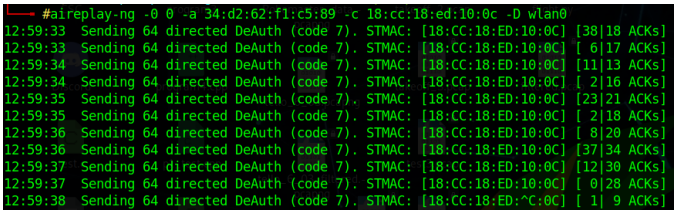


Fig. 2. Example of successful de-authentication attack using aireplay-ng command that is sending de-authentication frames to the UAV access point with a spoofed MAC address. Note: -0 specifies that we want to send de-authentication frames, 0 specifies that we want to send frames infinitely, -a is the MAC address of the access point, -c is the MAC address of the client we are spoofing, -D forces start without waiting for beacon, wlan0 is the name of the interface we are sending from.

and FDI attacks. A total of 40 flights with attack attempts were launched such that we have 10 flights for each attack.

1) De-authentication Attack: These attacks are the result of a vulnerability in the IEEE 802.11 protocol that allows for a de-authentication frame to be sent to an access point from a client ending the connection. This vulnerability can be exploited when an attacker sends a de-authentication frame to an access point using a spoofed address, which will then end the victim's connection with the access point. Furthermore, this is a generic attack that will work with any protocol (other than IEEE 802.11) that does not require any authentication [28]. For this attack, aireplay-ng was used to flood both the access point (the UAV) and the client (computer-1) with de-authentication frames.

Fig. 2 shows that the de-authentication packets are being sent to the communication channel associated with the UAV. This floods the channel between the UAV and the ground control station (computer-1), which results in terminating the connection between them. The communication can be disconnected for a finite or infinite duration depending on the attacker's goals. When computer-2 initiates a de-authentication attack, it sends a series of de-authentication frames to the UAV, making it believe that computer-1, the legitimate controller, has ended the session. This forces the UAV to drop the connection with computer-1 temporarily. The DJI Tello EDU is programmed to force land if no signal is received from a controller within 15 seconds, so this is generally what happens.

2) Replay Attack: Replay attacks can vary widely in functionality depending on the application. Generally, this attack involves capturing of communication between two parties to be replayed later in order to achieve some goal. Often, this attack is used to replay hashed passwords or the address resolution protocol (ARP) request to break the encryption between a host and client [29]. For UAVs, replay attacks can also be used to capture specific commands to be replayed by the attacker to effectively take control of the UAV [30].

To execute this attack, we utilized the aircrack-ng suite. We placed the ALFA network adapter into the monitor mode using the `airmon-ng start wlan0` command in order to capture wireless packets. We then initiated the packet capturing process with the `airodump-ng wlan0` command while ensuring the UAV was active and receiving legitimate commands from its controller. We saved the captured packets

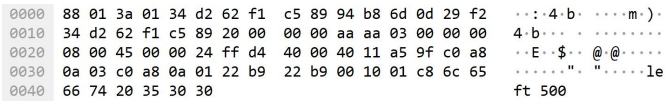


Fig. 3. Example command to move the UAV 500 cm to the left found in plain text via Wireshark. This command is captured and replayed by the attacker.

to a PCAP file and used Wireshark filters to examine the file and isolate the specific UAV command packets. We executed the command `aireplay-ng --inject replay -r capture.pcap wlan0`, where replay refers to replay mode, capture.pcap is the PCAP file containing the captured packets. This command replayed the captured command packets to the target UAV, deceiving it into re-executing the same commands it had previously received from the legitimate controller. Through replaying these command packets, the UAV showed abnormal behavior and was unable to properly follow its mission plan.

Fig. 3 depicts the Wireshark packet capturing interface. This shows the intercepted data between the UAV and controller (computer-1). During this attack, computer-2 captures valid data packets transmitted from computer-1 to the UAV. Once captured, computer-2 re-transmits these packets. During this window of attack, any commands sent by computer-1 to the UAV are effectively overwritten by the replayed packets, causing the UAV to execute the replayed commands instead.

3) Evil Twin Attack: To execute this attack, we utilized the Aircrack-ng and Airgeddon toolkits. The first step was to establish a rogue wireless access point that mimicked the legitimate service set identifier (SSID) of the target UAV's network. To obtain the actual SSID, the attacker ran the command `sudo airodump-ng wlan0` with its wireless interface in monitor mode as wlan0. This command listed the BSSIDs, SSID, and other network information. With the legitimate SSID acquired, we used the Airgeddon interface to create an evil twin access point, automatically configuring it with the cloned SSID of the target UAV network. To ensure the target UAV would connect to the attacker's rogue access point rather than the real one, the attacker needs to make its signal stronger. We accomplished this by employing the command `iwconfig wlan0 txpower 30` to boost the attacker's wireless adapter's transmission power to 30 dBm. Additionally, we briefly ran a de-authentication attack against the UAV to temporarily disconnect it from the legitimate access point, forcing it to reconnect and latch onto the stronger evil twin signal instead. Once connected to the attacker's rogue access point, the UAV became victim to a man-in-the-middle attack, allowing the attacker to monitor, intercept, steal, or manipulate data intended for the actual UAV network. In summary, through cloning the SSID and using a high transmission power, the attacker was able to successfully make the UAV connect to an evil twin access point and compromise its communications.

4) False Data Injection Attack: We utilized the Aircrack-ng suite, socket library, and custom Python scripts to execute an FDI attack against the UAV. To inject stealthy sensor readings and control commands, we first calculated the state-

space matrices (A, B, C) for the UAV using dynamic mode decomposition (DMD) and MATLAB's system identification toolbox. We focused our attack on three key states, namely, roll, pitch, and yaw. Next, we designed stealthy attack vectors with a custom Python script. Specifically, we modified the roll, pitch, and yaw measurements according to the equation $y = y + \gamma$, where y is the original sensor measurement and γ is the attack vector targeting the sensor data. Similarly, we modified the control signal as $u = u + \eta$, where u is the original control signal and η represents the attack vector for falsifying the command signal. We computed γ and η based on the (A, B, C) matrices to minimize residual errors for a stealthy attack. To inject the false data, we crafted packets using Python and sent them to the UAV's IP $192.168.10.1$ and controller port 8890. Consequently, the UAV received the falsified sensor and control data. This led to incorrect state estimation and control actions, causing irregular behavior as the UAV failed to maintain its planned mission.

Overall, the four attacks under consideration have clear manifestations in both cyber and physical domains. For instance, de-authentication attacks can disrupt the communication between the UAV and its controller, potentially leading to loss of control and UAV's hovering without completing its mission (i.e., cyber-attack leading to physical impact). Similarly, replay attacks can trick the UAV into accepting old or previously valid commands (cyber-attack), which lead to physical impact. Evil twin and FDI attacks follow the same rationale (cyber-attack with physical impact).

IV. Development of UAV Intrusion Detection System

This section discusses the development of an IDS model that fuses the cyber and physical features to yield improved attack detection performance. In the experimental results section, we will compare the IDS with cyber-only and physical-only IDSs that are trained on cyber or physical features to assess performance improvement.

A. Data Preparation

The raw data that we published in [8] and generated using the methodology described in Section III was split into cyber and physical datasets. Each dataset includes 75 files representing the recorded flights (35 for normal operation and 40 under cyber-attack). The cyber data was stored as PCAP files and the physical data was stored as numpy arrays. The physical data was directly entered into a pandas data frame; however, the cyber data had to be manually parsed first.

For the cyber data, each PCAP file was opened in Wireshark and filtered to only include data going to or from the BSSID associated with the Tello's access point. From here, each file was converted into JSON format and further parsed using a custom Python script to extract 36 cyber features plus an additional constructed feature that records the elapsed time since the last packet. At this point, both cyber and physical datasets were loaded into data frames and added an extra column indicating whether or not an attack occurred based on timestamps (i.e., assigned a class label). Within the cyber data frame, there were $32,978$ samples each with 37 features plus their associated class. Within the physical data frame, there were $6,236$ samples each with 16 physical features in addition to their associated class. The list of collected cyber and physical features is given in Table II.

TABLE II
Raw extracted features from both cyber and physical datasets

Cyber	Cyber	Physical
frame.number	ip.id	height
frame.len	ip.flags	x_speed
frame.protocols	ip.ttl	y_speed
wlan.duration	ip.proto	z_speed
wlan.ra	ip.src	pitch
wlan.ta	ip.dst	roll
wlan.da	tcp.srcport	yaw
wlan.sa	tcp.dstport	temperature
wlan.bssid	tcp.seq_raw	distance
wlan.frag	tcp.ack_raw	barometer
wlan.seq	tcp.hdr_len	flight_time
wlan.fc.type	tcp.flags	battery
wlan.fc.subtype	tcp.window_size	mp_distance_x
llc.type	tcp.options	mp_distance_y
ip.hdr_len	udp.srcport	mp_distance_z
ip.len	udp.dstport	timestamp_p
udp.length	data.data	
data.len	time_since_last_packet	
timestamp_c		

The following briefly explains the meaning of the features listed in Table II. For the 37 cyber features: frame.number denotes the number of each captured frame, frame.len is the length of the captured frame in bytes, frame.protocols field lists all the protocols encapsulated in the frame, wlan.duration denotes the time taken to transmit the frame on the wireless medium, wlan.ra is the receiver's MAC address, wlan.ta is the transmitter's MAC address, wlan.da is the destination MAC address, wlan.sa denotes the source MAC address, wlan.bssid is the MAC address of the access point, wlan.frag is the Wireless LAN fragment number, wlan.seq is the frame's sequence number, wlan.fc.type is the frame control type, wlan.fc.subtype is the subtype of the frame based on the frame control field, llc.type is the type field of the logical link control, ip.hdr_len is the length of the IP header in bytes, ip.len is the total length of the IP packet including header and data, ip.id is the identification field in the IP header, ip.flags indicates the flags in the IP header, ip.ttl is the time to live, ip.proto is the protocol used in the data portion of the IP datagram, ip.src is the source IP address, ip.dst is the destination IP address, tcp.srcport is the source port number for the TCP segment, tcp.dstport is the destination port number, tcp.seq is the sequence number of the first byte in the TCP segment, tcp.ack is the acknowledgment number in the TCP header, tcp.hdr_len is the length of the TCP header in bytes, tcp.flags indicates the flags in the TCP header, tcp.window size is the size of the receiver's window in bytes, tcp.options denotes optional settings for the TCP connection, udp.srcport is the source port number for UDP, udp.dstport is the destination port number for UDP, udp.length is the length of the UDP including header and data, data.data indicates the raw data of the captured frame, data.len is the length of the

data in the captured frame, and time_since_last_packet is the time elapsed since the previous packet was received. For the 16 physical features: height is the altitude of the UAV, x_speed, y_speed, and z_speed denote the speed of the UAV along the x, y, and z-axes, respectively, pith and roll denote the tilt of the UAV in the forward and backward direction, and the side to side, respectively, yaw is the rotation of the drone around the z-axis, temperature indicates the temperature reading from the UAV's motor, barometer measures the air pressure, flight time indicates the duration for which the UAV has been flying, the battery is the current battery level of the UAV, mp_distance_x, mp_distance_y, and mp_distance_z are the distances along the x, y, and z-axes from a Tello Pad, respectively. Finally, the timestamp_c and timestamp_p are timestamps for the collected cyber and physical features, respectively.

Due to the nature of how each dataset was recorded, these cyber-physical time series are asynchronous which makes combining them trickier than simply combining the features. The best way to combine these datasets was found to be by iterating through each unique timestamp and repeating the most recent features from either dataset. During this process, any data points collected before takeoff or after landing were discarded so as not to interfere with the learning process.

Next, both cyber and physical data frames were normalized using the min-max method to scale the data in the $[0, 1]$ range. This scaling is important for our data as it is a fusion of cyber and physical features with different units and scales. With normalization, faster convergence can be achieved during the IDS training. It should be noted that other normalization methods (e.g., standardization with zero mean and unit variance) can be used. Furthermore, we removed some features that were counterproductive. Those features were 'frame.number' so that the IDS models do not simply learn patterns in the numbering, 'barometer' because the atmospheric pressure has no correlation with cyber-attacks and could result in learning useless patterns, and finally, both cyber and physical timestamps were removed in order not to learn patterns in the time of day that attacks were launched. The resulting combined dataset was composed of $12,741$ benign samples and $16,843$ malicious samples. Data was split into train and test samples with a split ratio of $3:1$. For training and validation, 5-fold cross-validation was used.

B. Model Training and Optimization

To develop an effective UAV IDS, we investigated a range of machine learning models that represent: shallow and deep structures, and static and temporal learning models. We adopted SVM as a shallow model since it captures complex decision boundaries and offers particularly good performance at handling binary classification problems, making them suitable for intrusion detection tasks where the objective is to classify activities as normal or malicious. While deep learning models have gained attention in recent years, FNNs, being one of the foundational neural network architectures, offer a balance between complexity and performance. Finally, we studied models that exploit correlation within the data (LSTM-RNN and 1D-CNN models) to compare them against models that do not exploit correlation within the data (SVM and FNN). The adopted models have been considered in relevant state-of-the-art UAV IDS research, e.g., SVM [7], [16], FNN [21], LSTM [15], and CNN [14], [20].

1) IDS Based on SVM: This is an example of a supervised model that adopts a shallow structure and does not exploit the temporal correlation within time-series data. The aforementioned aspects make SVM a model that can be trained and deployed fairly easily. Four different kernels K were considered as hyper-parameters including linear, sigmoid, radial basis function (rbf), and polynomial. Also, different classifier margins C were considered, namely, $C \in \{0.1,\ 1,\ 10\}$. Finally, different scaling factor γ was considered, namely, $\gamma \in \{0.05, 0.1, 0.15, 0.2\}$. Random grid search was conducted using cross-validation with 5-folds to identify the optimal hyper-parameters for each model.

2) IDS Based on FNN: This model represents a deep structure that is trained in a supervised manner and handles static data, i.e., does not exploit the temporal correlation within the time-series data. Hyper-parameter optimization through a random grid search performed on validation data was conducted to determine the optimal structure of the model. The following search space was used while optimizing the hyper-parameters: number of hidden layers $L \in \{2, 3, 4, 5, 6, 8\}$, number of neurons $N \in \{64, 128, 256, 512, 1024\}$, activation function in the hidden layer $\varphi_{\text{h}} \in \{\text{Relu}, \text{tanh}\}$, activation function in the output layer $\varphi_{\text{o}} \in \{\text{sigmoid}, \text{softmax}\}$, and weight initialization function $I \in \{\text{random}, \text{glorot}, \text{he}\}$. The random grid search is used to identify the optimal hyper-parameters for each model.

3) IDS Based on LSTM-RNN: The LSTM-RNN model represents a deep structure that is trained in a supervised manner. It exploits the temporal correlation within our time-series data. The hyper-parameters of this model are optimized via random grid search. The hyper-parameter search space includes the following: number of hidden layers $L \in \{1, 2, 3, 4, 5\}$, number of LSTM cells $N \in \{64, 128, 256, 512, 1024\}$, activation function $\varphi \in \{\text{Relu}, \text{tanh}\}$, and weight initialization function $I \in \{\text{random}, \text{glorot}, \text{he}\}$.

4) IDS Based on 1D-CNN: This model also captures temporal correlations within the time-series data. This is achieved through convolutional filters that traverse across the sequential input data during model training and inference. By leveraging convolutions over the temporal dimension, the 1D-CNN can discern patterns and anomalies over time in the cyber-physical data streams. We identify the optimal hyper-parameters for each model using a random grid search. The hyper-parameter search space includes the following: number of convolutional layers $L \in \{2, 3, 4, 5, 6\}$, number of convolutional filters $F \in \{8, 16, 32, 64, 128\}$, kernel size $K \in \{2, 5, 10\}$, pooling size $P \in \{2, 3, 4, 5\}$, number of neurons in the dense layer $N \in \{32, 64, 128, 256, 512\}$, and dropout rate $r \in \{0.1, 0.2, 0.3, 0.4\}$.

C. Shap Analysis

SHAP analysis [31] was conducted on each dataset (cyber, physical, and combined/fused) and model. SHAP analysis uses

game theoretically optimal Shapley values to explain how various machine learning models make decisions. In our work, SHAP was used as a form of feature selection step informing our model on the most important features and conversely, the least important features that might need to be pruned. The optimal combinations of hyper-parameters found through grid search were used when running the SHAP analysis.

V. Experimental Results and Discussion

This section presents the detection results of the UAV cyber-physical IDS. For each machine learning model (SVM, FNN, LSTM, and 1D-CNN), three versions are developed and tested. These include versions trained on cyber-only, physical-only, and cyber-physical (fused/combined) datasets.

To evaluate the performance of the developed models (SVM, FNN, LSTM, and 1D-CNN) and their variants (cyber, physical, and cyber-physical), we conducted experiments involving two main assessment scenarios: complexity and range of attack training data. The first scenario reflects cases when models are trained on a single type of attack and tested against other unseen attacks. This demonstrates how models behave when trained on simple to advanced complex attacks. The second scenario reflects cases when the range of attacks considered in the training datasets increases and then the model is tested against other unseen attacks. When studying the impact of the complexity of attack training data, each model variant (cyber, physical, and cyber-physical) is trained on only one attack type and tested against all four attack types (including three unseen attacks). Therefore, each model undergoes 4 training and 16 tests (one training per attack, four tests per training), resulting in a total of 48 training and 192 tests for all models and all variants. When studying the impact of the range of attack training data, we considered two cases. First, models were trained on two attack types (evil twin and FDI) and tested on the combined four attacks. Second, models were trained on three attack types (evil twin, FDI, replay) and tested on the combined four attacks. Therefore, each model undergoes 2 training and 2 tests, resulting in a total of 24 training and 24 tests for all models and all variants.

We first show the SHAP analysis results to highlight the most effective features for the models. Then, for each model, we show the learning curves to examine whether the developed models are under or over-fitting. Finally, we compare the detection performance metrics for all the developed models in terms of model accuracy, precision, recall, F1 score, area under the curve (AUC) of the receiver operating characteristics (ROC). The definitions of these metrics are discussed next. First, all of the aforementioned metrics involve calculations using total True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). Accuracy gives an overall account of how many samples were correctly classified compared to the total classifications, which is given by

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{1}$$

Precision shows the comparison of correct positive predictions to total positive predictions, which can be described as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2}$$

Fig. 4. SVM-SHAP values for the top 20 features using the fused/combined cyber-physical data.

Recall shows the comparison of correct positive predictions to the total correct positive classifications, which is given by

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{3}$$

When dataset classes are uneven in size (i.e., imbalanced), as in this research and anomaly detection in general, these two metrics (Precision and Recall) generally offer more insights than accuracy. The F1 score provides a way to combine Precision and Recall into one metric using the harmonic mean through the formula

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{4}$$

A. Results of SHAP Analysis

We performed SHAP analysis using 50 random samples across each dataset and calculated their respective SHAP values. Due to space limitations, we are showing results for the SVM and FNN models using the fused cyber-physical data. The analysis resulted in very similar rankings of feature importance across all datasets (cyber-only, physical-only, and cyber-physical fusion) between the two models (SVM and FNN). Fig. 4 and Fig. 5 show the top 20 features selected by the SHAP analysis. All the models that are presented next were trained and tested on these features.

Out of the sixteen physical features listed in Table II, the following five physical features are deemed most effective: flight_time, temperature, battery, mp_distance_z, and mp_distance_x. Changes in these features reflect deviations in the expected UAV flight behavior, which could indicate potential intrusions. For instance, the flight duration (flight_time) is

impacted because the UAV's operation gets disrupted by some attacks (de-authentication and evil twin) or the UAV deviates from the original path (replay and FDI attacks). Also, the temperature and battery values change when an attack occurs as the attack could impact the flight duration hence affecting the UAV motor's temperature and battery. Similarly, the UAV's coordinates expressed by mp_distance_z and mp_distance_x could change due to a replay attack or FDI attack that causes the UAV to deviate from its normal path, etc.

Out of the thirty-seven cyber features listed in Table II, the following sixteen cyber features are deemed most effective: wlan.fc.type, wlan.duration, ip.hdr_len, ip.dst, frame.len, ip.proto, frame.protocols, wlan.seq, data.len, time_since_last_packet, ip.src, ip.ttl, ip.id, wlan.ra, llc.type, and udp.dstport. Launching an attack would affect the size of the frame, header, and data packets (frame.len, data.len, and ip.hdr_len) because the attack might introduce more/less data in the frame, data packet, and/or header. This also affects the connection time (wlan.duration and ip.ttl), e.g., in case of de-authentication and evil twin attacks. Also, the source and destination IP addresses, MAC addresses, and ports differ under attack from the benign case, e.g., in evil twin and de-authentication attacks, hence, affecting ip.src, ip.id, ip.dst, udp.dstport, and wlan.ra. Moreover, the packet transmission rate differs from the usual in some attacks (e.g., higher transmission rate for de-authentication attack as in Fig. 2, which is also manifisted in the evil twin attack), hence, affecting time_since_last_packet. The attack launching mechanism would also affect the adopted protocols and frame sequence number, hence, affecting ip.proto, frame.protocols, wlan.fc.type, llc.type, and wlan.seq.

As observed in Fig. 4 and Fig. 5, the SHAP analysis selected a combination of cyber and physical features, hence, underscoring the value of fusing both data types for robust UAV intrusion detection. Fusing cyber and physical data provides a more thorough representation of the UAV state, which in turn enhances the IDS's capability to detect attacks. Also, the SHAP analysis indicated more cyber features than physical ones. This gives an intuition that cyber-only models would outperform physical-only models, which will be proven in Subsection V.C.

B. Learning Curves for the Developed Models

The learning curves show the F1 scores on the training and validation datasets for the developed models. These curves are used to assess model fitting and ensure no under-fitting or over-fitting has occurred. The curves demonstrate the learning and generalization capabilities of the models.

In total, 216 learning curves were generated for all models and their variants under all scenarios. However, due to space constraints, we only include representative learning curves for each of the LSTM-RNN model variants (cyber, physical, and cyber-physical) to illustrate proper model fitting, as seen in Fig. 6. Using the SHAP-identified features and optimal hyper-parameters from grid search, we observe that the behavior of the training and validation F1 scores reflects good fitting. Notably, we employed a patience strategy of 10 epochs, where the model training was stopped if validation loss remained under 0.01 for 10 consecutive epochs.

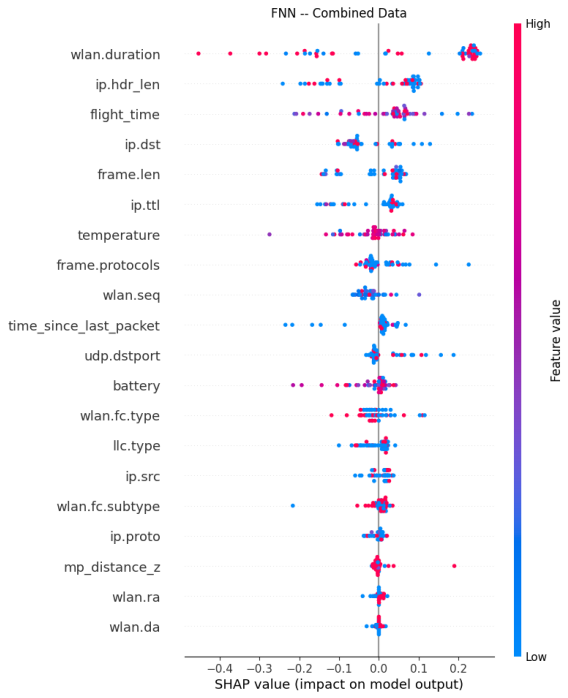


Fig. 5. FNN-SHAP values for the top 20 features using the fused/combined cyber-physical data.

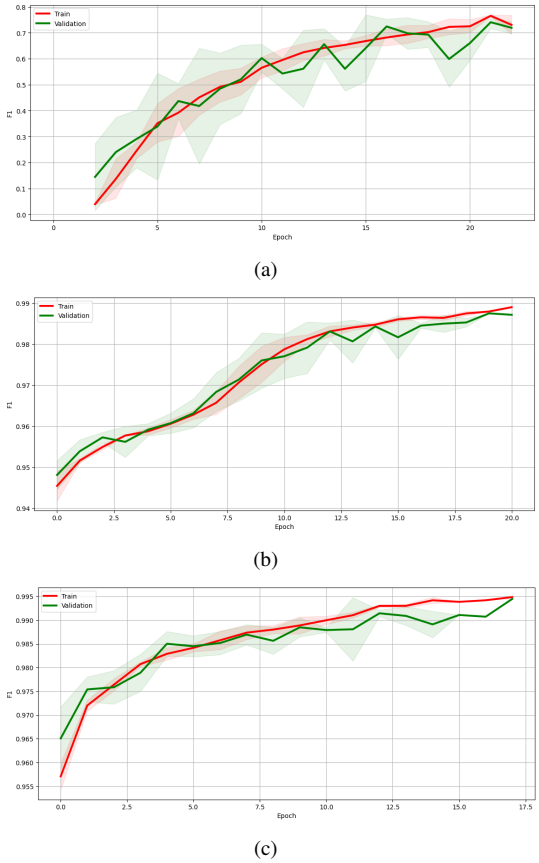
The other models (SVM, FNN, and 1D-CNN) exhibit similar performance to Fig. 6 for all variants (cyber-only, physical-only, and cyber-physical) indicating a good fitting model with no under-fitting or over-fitting.

C. Performance Results

In the following, we show the results for the two scenarios, namely, the complexity and range of attack training data.

1) Complexity of Attack Training Data: In this scenario, we examine model performance when trained on a single type of attack (de-authentication, replay, evil twin, or FDI attacks) and evaluated against the other three (unseen) attacks and the attack type adopted during the model training. Hence, the attack training data ranges in complexity from the relatively simple de-authentication attack to the more sophisticated stealthy FDI.

To elaborate, the evaluation procedure is as follows. Each model variant is trained on only one attack type, such as SVM cyber-only trained solely on the de-authentication attack. The trained model is then tested against unseen attacks, i.e., replay, evil twin, and FDI attacks, along with the de-authentication attack. This process is repeated for each attack type totaling four unique trainings per model. For each training, there are four test scenarios: the three unseen attacks and the attack type used for training. Therefore, each model undergoes 4 trainings and 16 tests (one training per attack, four tests per training), resulting in 48 total trainings and 192 tests. However, due to space limitations, we provide the average performance per model variant. For example, for the SVM cyber-only model trained on the de-authentication attack, we average its



(a)

(b)

(c)

Fig. 6. Learning curves for LSTM model trained on (a) physical-only, (b) cyber-only, and (b) fused/combined cyber-physical data.

performance on testing against replay, evil twin, FDI, and de-authentication attacks.

The detection performance for cyber-only, physical-only, and cyber-physical variants are shown in Fig. 7, Fig. 8, and Fig. 9, respectively. The x-axis show the attack training data and the y-axis gives the average detection metric over all attacks. The following observations can be made:

- Fig.7 presents the average performance for models trained and tested on cyber-only data. Across all models, training on the simple de-authentication attack leads to the lowest performance in detecting unseen attacks. In contrast, training on the complex FDI attack results in the highest model performance against unseen attacks. The 1D-CNN architecture achieves the highest average F1-score of 76.82% when trained on FDI attacks.
- Fig. 8 shows average performance for models trained and tested on physical-only data. Training on de-authentication attacks again leads to the lowest generalization ability. The 1D-CNN model demonstrates the highest performance with an average F1-score of 63.82% when trained on FDI attacks.
- Fig. 9 presents average results for models trained and tested on combined cyber-physical data. The 1D-CNN achieves the highest average F1-score of 80.33% when trained only on the complex FDI attack.
- The results demonstrate that models achieve higher detection performance against unseen attacks when trained on complex attack data such as FDI. The simple de-authentication attack patterns fail to properly generalize the model's ability to detect other unseen attacks.
- Models trained on physical-only features offered the worst performance. On the other hand, models trained on cyber-physical features outperformed the models trained on cyber-only and physical-only features. Specifically, an improvement of $1-4\%$ is observed in the F1 score across all models when trained on cyber-physical features versus training on cyber-only features. Similar behavior is consistent among all other metrics.

2) Range of Attack Training Data: The goal of this assessment is to examine the model performance when the range of attack data increases during the training. We consider two cases. In the first case, models were trained on two attack types, evil twin and FDI, and tested against all four attack types. In the second case, the models were trained on three attack types (evil twin, FDI, and replay) and tested on all four attack types. The detection performance results are shown in Fig. 10 and Fig. 11, for the two cases, respectively. The following observation can be made:

- Fig. 10 shows the detection results when models are trained on two attacks (evil twin and FDI attacks) and tested on all four attacks. The results here are consistent with those in Fig. 7 - Fig. 9 where the cyber models outperform the physical models, and cyber-physical performance was the best with $2-4\%$ improvement over the cyber-only models. Again, the 1D-CNN model trained on cyber-physical has the highest F1-score i.e., 94.11%.
- Fig. 11 shows results when models are trained on the dataset from three attacks (replay, evil twin, and FDI attacks) and tested on all four attacks. It can be seen clearly that cyber-physical models outperform cyber-only and physical-only models, with an improvement of $1-3\%$ over the cyber-only models. The 1D-CNN model offers the highest detection performance with 96.13% F1-score.
- It can be observed that expanding the range of attacks during the training stage offers an improvement of $2-7\%$ in F1 score. Similar behavior is observed for all other metrics. Systematically increasing the range of attacks in the training data helps the model generalize and improves the detection ability.

D. Discussions

The fusion of cyber and physical data provides a comprehensive representation of a UAV's operational state. While cyber data captures anomalies in communication patterns, physical data reveals discrepancies in flight dynamics and sudden changes in behavior such as roll, pitch, yaw angles, acceleration, etc. By fusing these data streams, we construct a detailed depiction of the UAV's state across cyber and physical domains. This fusion enables accurate detection of attacks that might be missed when only one type of data is considered.

To improve the IDS generalization ability against unseen attacks, the model should be trained on complex attack patterns and a wide range of attacks.

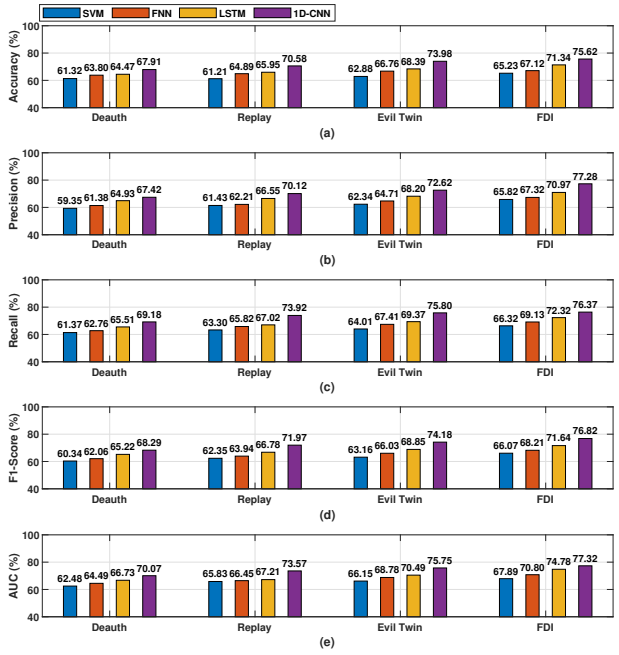


Fig. 7. Average detection performance for models trained on cyber-only features for the first scenario - complexity of attack training data.

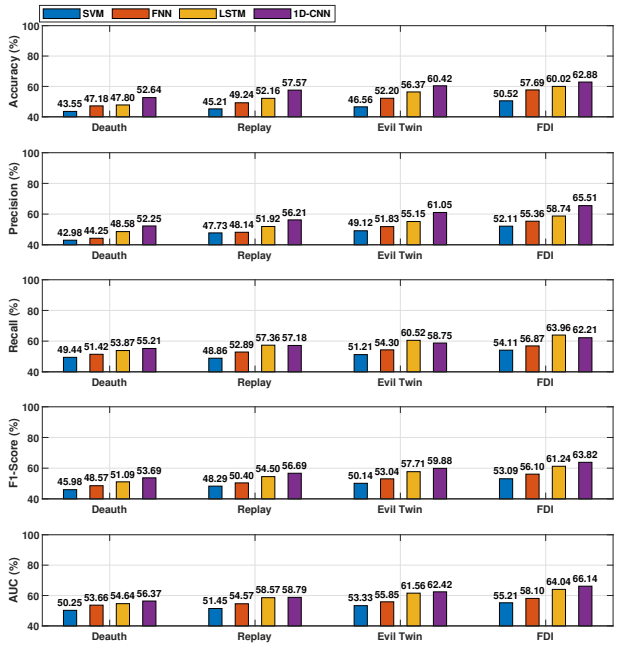


Fig. 8. Average detection performance for models trained on physical-only features for the first scenario - complexity of attack training data.

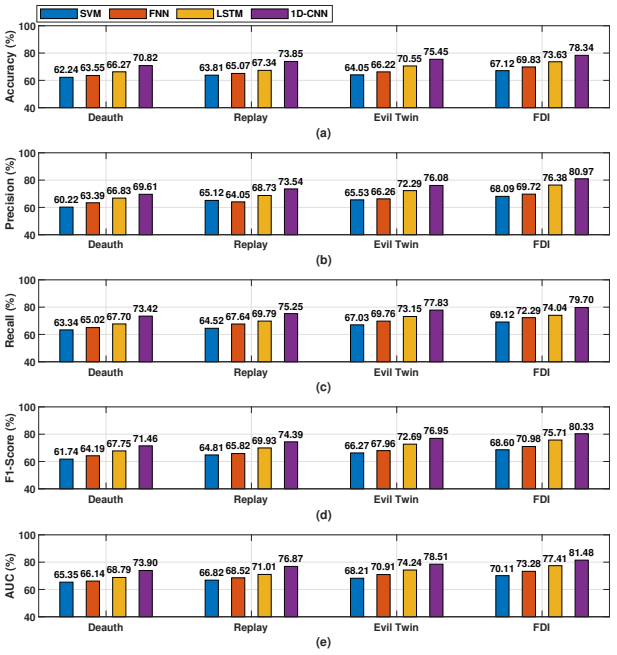


Fig. 9. Average detection performance for models trained on cyber-physical features for the first scenario - complexity of attack training data.

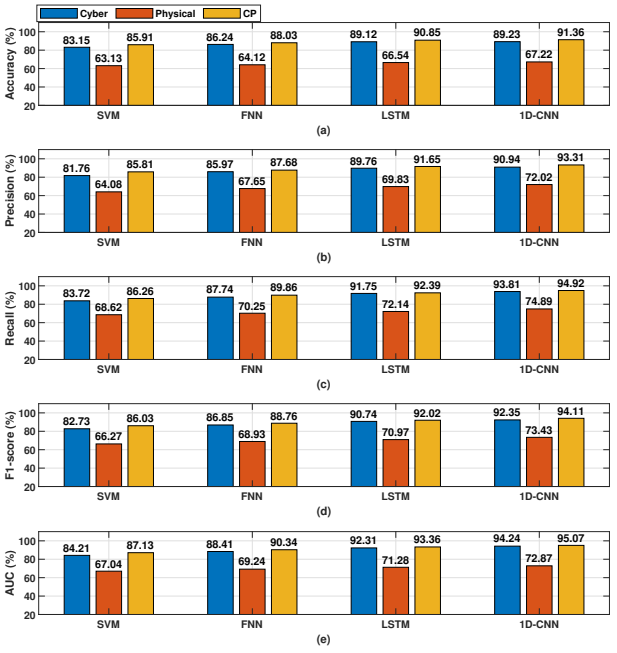


Fig. 10. Detection performance for models trained on two attacks and tested on all four attacks. CP is cyber-physical.

VI. Conclusion and Future Work

In this paper, we developed a testbed that involves a UAV, controller, computers, and antenna for data collection. The testbed was used to create a practical published dataset that contains both cyber and physical features of UAV operation under normal and cyber-attack conditions. The dataset was used to train and test machine learning-based IDSs (SVM, FNN, LSTM-RNN, and 1D-CNN). We performed extensive experiments and carried out two evaluation scenarios that assessed the IDS performance for various attack complexity and range in the training data. Our experimental results demonstrated that (a) cyber-physical fusion can improve the detection performance in both evaluation scenarios when compared with models that are trained on cyber-only or physical-only

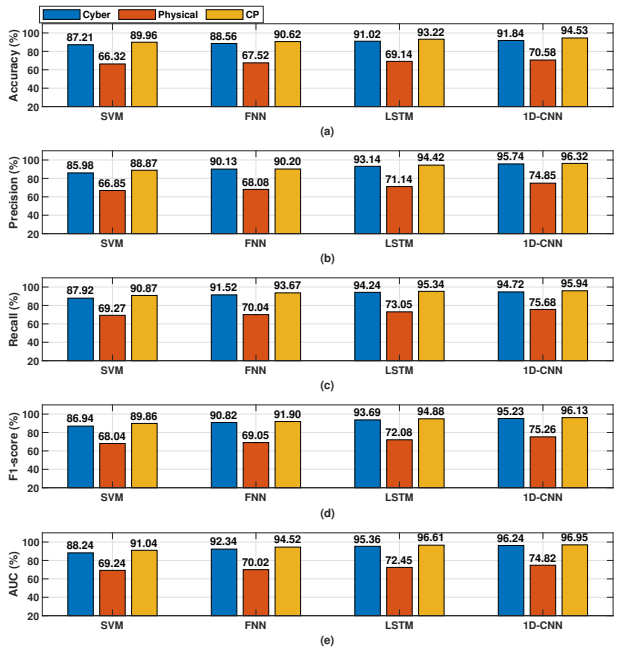


Fig. 11. Detection performance for models trained on three attacks and tested on all four attacks. CP is cyber-physical.

datasets, and (b) training IDS on a complex-wide range of attacks helps improve its generalization ability against unseen attacks that were not in the training set.

In our future work, we will incorporate more UAVs in the system and investigate IDS for a swarm of UAVs.

REFERENCES

[1] F. Al-Turjman, M. Abujubbeh, A. Malekloo, and L. Mostarda, "UAVs assessment in software-defined IoT networks: An overview," *Computer Communications*, vol. 150, pp. 519–536, 2020.

[2] B. Chen, D. W. Ho, G. Hu, and L. Yu, "Secure fusion estimation for bandwidth constrained cyber-physical systems under replay attacks," *IEEE transactions on cybernetics*, vol. 48, no. 6, pp. 1862–1876, 2017.

[3] O. Westerlund and R. Asif, "Drone hacking with raspberry-pi 3 and wifi pineapple: Security and privacy threats for the internet-of-things," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*. IEEE, 2019, pp. 1–10.

[4] J. Noh, Y. Kwon, Y. Son, H. Shin, D. Kim, J. Choi, and Y. Kim, "Tractor beam: Safe-hijacking of consumer drones with adaptive GPS spoofing," *ACM Transactions on Privacy and Security (TOPS)*, vol. 22, no. 2, pp. 1–26, 2019.

[5] H. Sedjelmaci, S. M. Senouci, and N. Ansari, "Intrusion detection and ejection framework against lethal attacks in UAV-aided networks: A bayesian game-theoretic methodology," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1143–1153, 2016.

[6] R. Zhang and et. al., "Design of a novel network intrusion detection system for drone communications," in *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. IEEE, 2018, pp. 1–10.

[7] M. A. Ferrag and L. Maglaras, "Deliverycoin: An IDS and blockchain-based delivery framework for drone-delivered services," *Computers*, vol. 8, no. 3, p. 58, 2019.

[8] S. Hassler, U. Mughal, and M. Ismail, "Cyber-physical dataset for UAVs under normal operations and cyber-attacks," 2023. [Online]. Available: https://dx.doi.org/10.21227/6f22-py65

[9] X. Tan, S. Su, Z. Zuo, X. Guo, and X. Sun, "Intrusion detection of UAVs based on the deep belief network optimized by PSO," *Sensors*, vol. 19, no. 24, p. 5529, 2019.

[10] S. Ouiazzane and et. al., "Towards a multi-agent based network intrusion detection system for a fleet of drones," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 10, 2020.

[11] R. Shrestha, A. Omidkar, S. A. Roudi, R. Abbas, and S. Kim, "Machine-learning-enabled intrusion detection system for cellular connected UAV networks," *Electronics*, vol. 10, no. 13, p. 1549, 2021.

[12] V. Praveena, A. Vijayaraj, P. Chinnasamy, I. Ali, R. Alroobaea, S. Y. Alyahyan, and M. A. Raza, "Optimal deep reinforcement learning for intrusion detection in UAVs," *CMC-Computers Materials & Continua*, vol. 70, no. 2, pp. 2639–2653, 2022.

[13] O. Bouhamed, O. Bouachir, M. Aloqaily, and I. Al Ridhawi, "Lightweight IDS for UAV networks: A periodic deep reinforcement learning-based approach," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2021, pp. 1032–1037.

[14] L. Kou, S. Ding, T. Wu, W. Dong, and Y. Yin, "An intrusion detection model for drone communication network in sdn environment," *Drones*, vol. 6, no. 11, p. 342, 2022.

[15] J. Han and W. Pak, "Hierarchical LSTM-based network intrusion detection system using hybrid classification," *Applied Sciences*, vol. 13, no. 5, p. 3089, 2023.

[16] R. T. Mehmood, G. Ahmed, and S. Siddiqui, "Simulating ML-based intrusion detection system for unmanned aerial vehicles (UAVs) using COOJA simulator," in *2022 16th International Conference on Open Source Systems and Technologies (ICOSST)*. IEEE, 2022, pp. 1–10.

[17] K. H. Park, E. Park, and H. K. Kim, "Unsupervised fault detection on unmanned aerial vehicles: Encoding and thresholding approach," *Sensors*, vol. 21, no. 6, pp. 1–17, 2021.

[18] E. Basan, A. Basan, A. Nekrasov, C. Fidge, J. Gamec, and M. Gamcová, "A self-diagnosis method for detecting UAV cyber attacks based on analysis of parameter changes," *Sensors*, vol. 21, no. 2, p. 509, 2021.

[19] E. Basan, A. Basan, A. Nekrasov, C. Fidge, N. Sushkin, and O. Peskova, "GPS-spoofing attack detection technology for UAVs based on kull-back–leibler divergence," *Drones*, vol. 6, no. 1, 2022.

[20] H. Ahn, H.-L. Choi, M. Kang, and S. Moon, "Learning-based anomaly detection and monitoring for swarm drone flights," *Applied Sciences*, vol. 9, no. 24, p. 5477, 2019.

[21] E. M. Khanapuri, R. Sharma, and K. Brink, "Learning-based detection of stealthy false data injection attack applied to cooperative localization problem," in *AIAA SCITECH 2022 Forum*, 2022, p. 2543.

[22] DJI, "Tello edu." [Online]. Available: https://www.ryzerobotics.com/tello-edu

[23] ALFA Network, "Awus036ach." [Online]. Available: https://www.alfa.com.tw/products/awus036ach?variant=36473965871176

[24] Lorenzo "Palinuro" Faletra, Parrot Dev Team, "Parrotos security." [Online]. Available: https://www.parrotsec.org/

[25] Thomas d'Otreppe de Bouvette, "Aircrack-ng." [Online]. Available: https://www.aircrack-ng.org/

[26] The Tcpdump team, "tcpdump." [Online]. Available: https://www.tcpdump.org/

[27] Wireshark Foundation, "Wireshark." [Online]. Available: https://www.wireshark.org/

[28] J. Bellardo and S. Savage, "802.11 denial-of-service attacks: Real vulnerabilities and practical solutions," in *12th USENIX Security Symposium (USENIX Security 03)*. USENIX Association, aug 2003.

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

[30] D. Rudo and K. Zeng, "Consumer UAV cybersecurity vulnerability assessment using fuzzing tests," 2020.

[31] Scott Lundberg, "Shapley additive explanations." [Online]. Available: https://shap.readthedocs.io/en/latest/index.html