A Light Boosting-based ML Model for Detecting Deceptive Jamming Attacks on UAVs

Hadjar Ould Slimane, Selma Benouadah, Tala Talaei Khoei, and Naima Kaabouch
School of Electrical Engineering and Computer Science,
University of North Dakota, Grand Forks, ND 58202 USA

2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC) | 978-1-6654-8303-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CCWC54503.2022.9720830

***Abstract*—Advances made in Unmanned Aircraft Vehicles (UAVs) have increased rapidly in the last decade resulting in new applications in both civil and military spheres. However, with the growth in the usage of these systems, various cybersecurity challenges arose unveiling the vulnerabilities of UAV wireless networks. Among the attacks that threaten the network's availability and reduce their performance are jamming attacks. Several approaches have been proposed to address this problem; however, most of them are not suitable for UAVs due to their reduced size, weight, and power constraints. In this paper, we propose a lightweight machine learning technique, LightGBM, to detect deceptive jamming attacks on UAV networks. The performance of this model is compared to that of three boosting and bagging-based machine learning models namely, XGBoost, Gradient Boost, and Random Forest. The results show that, although the LightGBM model has slightly lower accuracy (98.4%) than Gradient Boost (99%) and Random Forest (98.87%), it is 21 times faster and occupies two times less memory during the prediction than Gradient Boost and Random Forest.**

***Index Terms*—UAV, jamming attacks, detection techniques, machine learning, LightGBM, deceptive jamming, wireless networks.**

I. Introduction

Over the last decade, the use of Unmanned Aerial Vehicles (UAV) in various domains has been remarkably increasing, making them a potential target to a myriad of attacks. In addition, some of the applications may be in sensitive sectors in which a breach in security can lead to disastrous consequences. Furthermore, wireless communication is established between a UAV and a Ground Control Station (GCS) for command and control, which can open the venue to a variety of threats, including malicious users who can easily access vital information or disturb the communication systems [1].

The data communications between the UAV network nodes can be categorized into four distinct types: data link between a UAV and another UAV, UAV and a GCS, UAV and the Global Positioning System (GPS), and UAV and the Automatic Dependent Surveillance-Broadcast (ADS-B) ground stations. Each component and data link is susceptible to a range of risks that can degrade the performance of the UAV and damage the network.

Jamming attacks are one of the most serious threats to UAV networks. A jamming attack is a Denial of Service (DoS) threat that compromises network availability. In this attack, the jammer transmits radio signals to flood the channel and disrupt ongoing communications [2]. For this purpose, several papers have been published, proposing techniques to detect jamming attacks on UAVs. For example, in [3], the authors presented an anomaly-based Intrusion Detection System (IDS) to detect GPS spoofing and jamming attacks on UAV networks. This IDS is composed of a Self-Taught Learning algorithm and a multiclass Support Vector Machine (SVM) algorithm. The results show that this approach has an accuracy of 90%. In [4], the authors proposed an intrusion detection and response that monitor the behaviors of a UAV. The intrusion detection technique checks if the number of sent packets and the jitter exceeds a certain threshold to detect the constant and deceptive jamming attacks. This model provides a detection rate of more than 93%. In [5], a framework is proposed to detect jamming attacks in UAV networks using two Machine Learning (ML) models, multi-layer perceptron, and decision trees that reach an accuracy of 96%. In [6], the authors presented a localization-based approach to estimate the jammer's location range and transmission power. Another method is discussed in [2] that uses federated learning and a Dempster-Shafer-based client selection approach to detect jamming attacks in flying ad hoc networks. A semantic analysis-based approach was suggested in [7], where the state of the UAV is monitored to detect changes using fuzzy logic.

Although these approaches achieve satisfactory results in terms of accuracy, either they simulated only simple jamming attacks making them ineffective in detecting deceptive jamming attacks, or require additional hardware making them costly to deploy. Furthermore, most studies focused on generic wireless communication jamming attacks [8]–[13]. The techniques used in those studies are impractical solutions for UAVs due to their limited power, low computational resources, and high mobility. Therefore, motivated by the limitations of the existing studies, we propose a fast, accurate, and lightweight ML model to detect deceptive jamming attacks that target UAV networks.

The main contributions of this paper can be summarized as follows:

- Investigation and selection of the most important features in detecting deceptive jamming attacks.
- A fast, lightweight, and high-performance machine-learning model that can be implemented in a UAV to locally identify jamming attacks.

- A comparative analysis of a fast and lightweight ML model, LightGBM, with other traditional ML models.

This paper's remainder is organized as follows: Section II discusses the methodology, including the jamming scenario, data pre-processing, feature selection, and machine learning models used to conduct this research. In Section III, we present the simulation results and discuss them. Finally, Section IV provides a conclusion and recommendations for future works.

II. Methodology

This section discusses the jamming scenario as well as the features of the dataset used in this study. In addition, as shown in Fig.1, we briefly highlight the main techniques utilized for data pre-processing, feature selection, ML classification models, and hyperparameter tuning techniques.

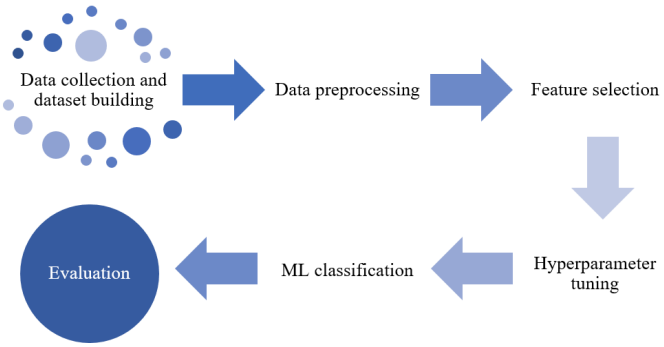

Fig. 1. Jamming attack detection ML model workflow diagram

A. Jamming Scenario

The drone and the GCS communicate using wireless communications over a 2.4 GHz or 5 GHz channel, such as Bluetooth or Wi-Fi 802.11 [14]. In this study, we used a Simulink model of Wi-Fi 802.11 to simulate the communication between the UAVs and the GCS. We considered two scenarios, a normal and a jamming attack case. For the normal case, we use three nodes, two UAVs and the GCS. The jamming attack case consists of a UAV, a jammer, and a GCS.

We used another UAV in the first scenario to ensure a fair comparison between the two scenarios for detecting jamming attacks, as having another UAV may result in packet collision, which might be interpreted as jamming in traditional jamming detection techniques. According to the Federal Aviation Administration (FAA), UAVs fly at the same altitude throughout most of their flight [8], therefore only two dimensions were considered in this work. The UAVs' movements were simulated as a distance ranging between 30 and 70 meters. The details of the used channel are shown in Table I.

In this study, we implemented a deceptive jammer since it is one of the most stealthy jammers compared to other types, such as constant and random jammers [15]. It uses a low level of power within the range of legitimate nodes' powers and broadcasts a continuous stream of data to deceive these nodes into believing it is transmitting authentic data [16]. As a result, traditional methods of jamming attacks detection are ineffective in detecting this type of jammers. For example, the Received Signal Strength (RSS) does not change since the jammer uses low power, which is similar to the power coming from legitimate nodes.

TABLE I. Channel Parameters

Parameter	Value
Bandwidth	5 GHz
Data Rate	12 Mbps
Modulation	QPSK
Channel impairments	Free-space path loss, range propagation loss, and Rayleigh fading

Table II shows the parameters used for simulating the normal and the jammer nodes. The deceptive jamming was performed by reducing the inter-packet interval to 3-6 milliseconds to have a continuous data transmission. This process occupies the channel for an extended period of time, preventing other nodes from transmitting data.

TABLE II. Simulation Parameters

Parameter	Normal node	Jammer node
Transmitting power	5-10 dBm	5-10 dBm
Antenna gain	1 dB	1 dB
Inter packet interval	10-15 ms	3-6 ms
Packet size	1000 Bytes	1000 Bytes

B. Data Pre-Processing

The extracted dataset for this study contains 10,000 samples with 25 features from several communication layers; 5000 samples were extracted from jamming signals and 5000 samples were extracted from regular traffic. Following the acquisition of raw data, we applied data pre-processing techniques to get clean data ready to be fed to our machine-learning model and provide better results. We encoded the data into 0 as legitimate and 1 as malicious. After the encoding, we employed the feature scaling technique, min-max scaling, to normalize the corresponding data. The features are rescaled to a range between 0 and 1 in this approach.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

Where X' denotes the normalized feature value, X_{min} and X_{max} are the minimum and maximum values of the selected feature, respectively, and X is the feature value.

C. Feature Selection

Feature selection techniques are one of the most important aspects of building good ML models. These strategies identify the most important features and show the correlated features in the dataset. As some of the features might be ineffective in the models' training thus they increase the model time and complexity, and reduce the accuracy. Therefore, it is important

to identify the most important features to guarantee optimal results. In this study, we used two feature selection techniques, Spearman Correlation and Extra Tree classifier. We will go through these techniques in further depth in the sections that follow.

1) Spearman Correlation: In this work, we applied the Spearman correlation method to find then remove correlated features. This technique was chosen due to its ability to measure the monotonic relation between two variables in addition to its higher robustness to outliers than other feature selection techniques [17]. This method demonstrates how strong the relationship is between two variables through the Spearman rank correlation coefficient defined as follows:

$$r = \frac{Cov(X_r, Y_r)}{Std(X_r)Std(Y_r)} \tag{2}$$

Where X_r and Y_r are the ranked values of the random variables X and Y respectively; $Cov()$ is the covariance and $Std()$ is the standard deviation. In this paper, we consider that two variables are correlated if r provides results higher than 0.9 or lower than -0.9.

2) Extra Tree Classifier: To reduce the computational complexity and processing time, we used the Extra Tree classifier to select the most important features. The Extra Tree, also known as Extremely Randomized Tree [18], is a tree-based ensemble ML method similar to the Random Forest method with a simpler algorithm. The predictions are calculated by averaging the results from the decision trees. The use of the averaging combined with the randomization in the trees' construction helps improve the predictive accuracy, reduce variance, and control over-fitting [18] to evaluate feature importance, which can be exploited for feature selection.

D. Machine Learning Models

The purpose of this research is to build a fast, efficient, and lightweight model that detects jamming in UAV networks. This can be considered as a binary classification problem, where the machine-learning model categorizes the data into jammed (1) or not jammed (0).

In the last few years, ensemble learning algorithms have shown impressive abilities to solve classification problems compared to conventional learning algorithms [19]. The aim of these techniques is to combine the predictions of multiple weak learners. In addition, these algorithms have a higher generalization power than that of their member learners, which leads to much more accurate results. Ensemble learning methods can be divided into three main classes: bagging, boosting, and stacking.

In this work, we used the LightGBM algorithm, which is a tree-based gradient boosting model [20]. While the Gradient Boost algorithm is known for its efficiency and accuracy, LightGBM has a faster training speed and lower memory usage and can handle large-scale datasets. Unlike Gradient Boost or XGBoost, which processes all the data samples to compute the information gain of all possible data splits, the LightGBM uses two methods to reduce its complexity: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB).

GOSS considers data instances with large gradients and applies random sampling on the data samples with small gradients. On the other hand, EFB is used to reduce the number of features effectively. This technique bundles exclusive features into a single feature called exclusive feature bundle. This speeds up the process and reduces the data size without affecting the accuracy [20].

1) Hyperparameter Tuning: In the LightGBM model, as in any machine-learning model, hyperparameter tuning constitutes a key step in optimizing the results of the ML model. Several hyperparameter optimization methods exist in the literature, from grid search to random search to more advanced approaches like Bayesian optimization, genetic algorithms, and artificial neural networks. In our study, we used grid search to find the optimal combination of hyperparameters. This technique is one of the most basic ones and is used generally with new models and datasets as the hyperparameters' importance and impact are yet to be determined and analyzed. Since LightGBM is relatively still a new model, the grid search method is considered the best one to find the optimal hyperparameters

III. Results and Discussion

In this work, 5-fold cross-validation was used to train 70% of the data and test 30% of the remaining dataset. We initially identified 25 features: medium access control (MAC) transmission fails (MAC Tx Fails), MAC frame retries (MAC Retries), dropped received physical signals (Phy Rx Drop), dropped received MAC frames (MAC Rx Drop), transmitted MAC frames in bytes (MAC Tx Bytes), successfully transmitted MAC frames (MAC Tx Success), idle state time, contend state time, sending data state time, wait for Rx state time, extended inter-frame spacing (EIFS) state time, Rx state time, throughput, MAC average time per frame, MAC max queue length, MAC back off, Rx triggers while previous Rx is in progress, Rx triggers while Tx in progress, signal power, signal to noise ratio (SNR), root mean square of error vector magnitude (RMS EVM), packet delivery ratio (PDR), bad packet ratio (BPR), eye height, and eye width.

As shown in Fig. 2, there are 11 pairs of correlated features; therefore, 11 features were removed leaving14 features in the list. To balance between the used computational power and the detection efficiency only the features with importance higher than 0.05 were selected using the Extra Tree classifier, namely Rx Time, Idle Time, and BPR. SNR, and Signal Power, which have an importance of approximately 0.6, 0.18, 0.07, 0.05, and 0.05, respectively, as illustrated in Fig. 3. The selected features are defined as:

Signal to Noise Ratio (SNR): SNR is defined as the received signal power over the noise power.

$$SNR = \frac{Signal\ Power}{Noise\ Power} \tag{3}$$

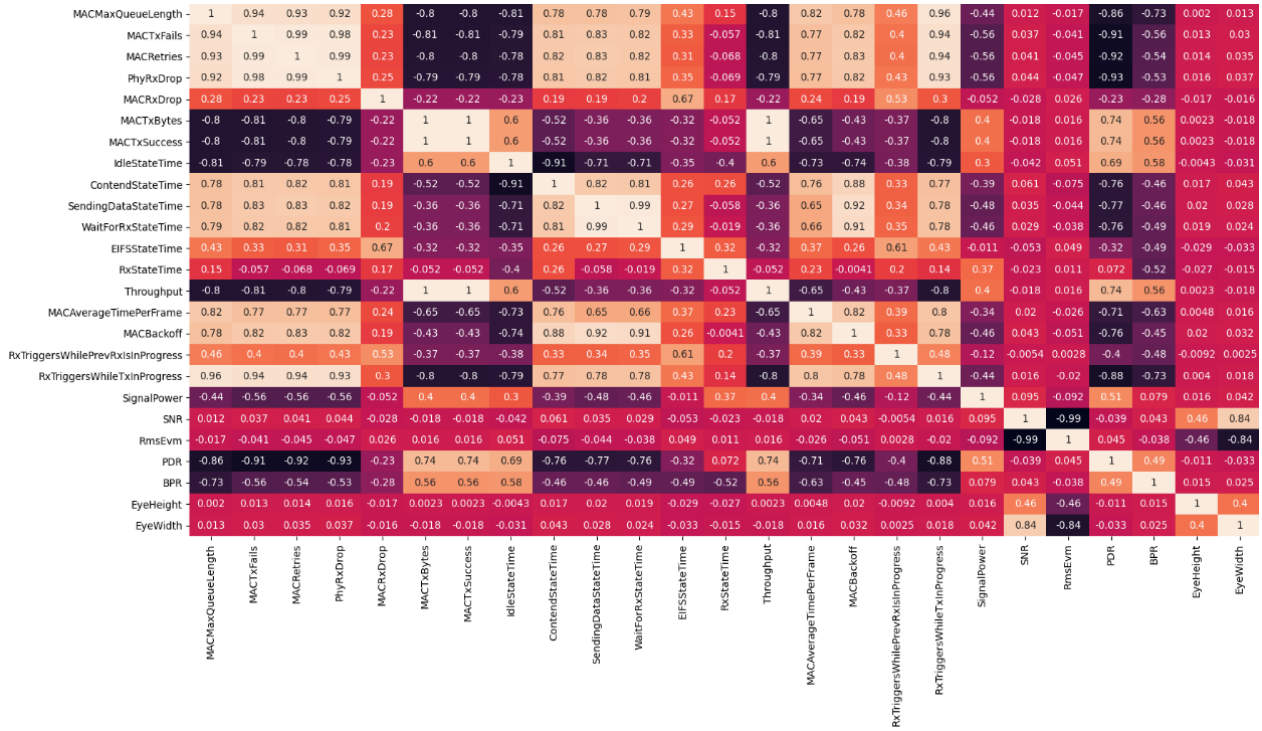

Fig. 2. Heatmap of Spearman's feature correlation

Bad Packet Ratio (BPR): BPR is the number of corrupted received packets that failed the Cyclic Redundancy Check (CRC) over the total number of received packets.

$$BPR = \frac{Number\ of\ Corrupted\ Received\ Packets}{Total\ Number\ of\ Received\ Packets} \quad (4)$$

Signal Power: Signal power is the power of the received signal that surrounds the reception antenna, and it is measured in dBm.

Rx Time: It is the total time that a node spends in receiving the data after the Clear Channel Assessment (CCA) has been sent. Frames are received and processed in this period, and it is expressed in microseconds.

Idle Time: It is the total time between two consecutive successful packets, and it is expressed in microseconds.

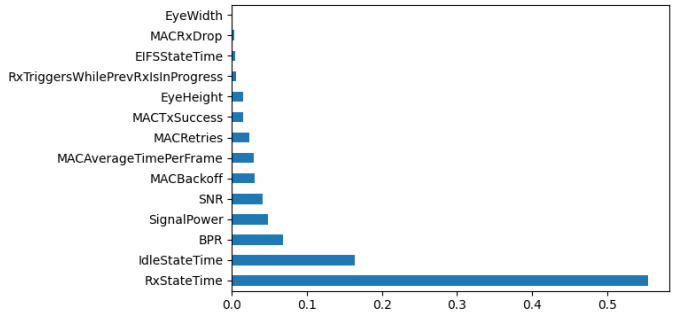

Fig. 3. Extra Tree classifier features importance

Next, we compared the performance of the LightGBM model with those of Random Forest, Gradient Boost, and XGBoost in terms of probability of detection (TPR), probability of misdetection (FNR), probability of false alarm (FPR), accuracy (ACC), processing time, complexity, and memory size. These metrics are defined as follows:

The probability of detection refers to the probability of correctly classifying malicious signals as jamming attacks over the total number of jamming signals.

The probability of misdetection gives the percentage of the number of jamming signals that were classified as legitimate signals over the total number of jamming signals.

The probability of false alarm is the percentage of legitimate signals that were classified as jamming signals over the total number of non-jamming signals.

Accuracy is the total number of correctly classified jamming and legitimate signals over the total number of received signals.

Processing time is the execution time of the ML algorithm during the training period.

Complexity is the computational complexity that is driven by the ML model algorithm. It describes the amount of time to run the program.

Memory size is the memory usage of the ML model during the training and the prediction period.

Where TP stands for the number of true positives, FP for false positives, TN for true negatives, and FN for false negatives. All the selected classification models' hyperparameters were chosen based on the grid search method to obtain a fair comparison between the models.

Table III shows the optimum values of ML model hyperparameters in terms of evaluation metrics derived using the

grid search tuning approach. The accuracy and probabilities of detection, misdetection, and false alarm of the selected ML models are illustrated in Fig. 4 and Fig. 5.

TABLE III. Hyperparameters of ML Models

ML Model	Hyperparameters
Random Forest	max depth= 7, learning rate= 0.2, colsample by tree= 0.6, subsample= 0.9, lambda= 5.
Gradient Boost	max depth= 3, n estimators= 150, min sample leaf= 6, max features= 5.
XGBoost	max depth= 7, learning rate= 0.2, colsample by tree= 0.6, subsample= 0.9, lambda= 5, gamma= 0.25.
LightGBM	max depth= 7 ,learning rate= 0.1, num leaves= 100, max bin= 100.

It can be observed that all models exhibit relatively high performance in terms of probability of detection, accuracy, misdetection, and false alarm. However, it is clear that the Gradient Boost shows the best results with a detection probability and an accuracy of 99% while the misdetection and false alarm probabilities are both below 1%. This algorithm is followed by Random Forest, which is only 0.13% less than Gradient Boost in the detection probability and accuracy and is higher in the probabilities of misdetection and false alarm by less than 0.3%.

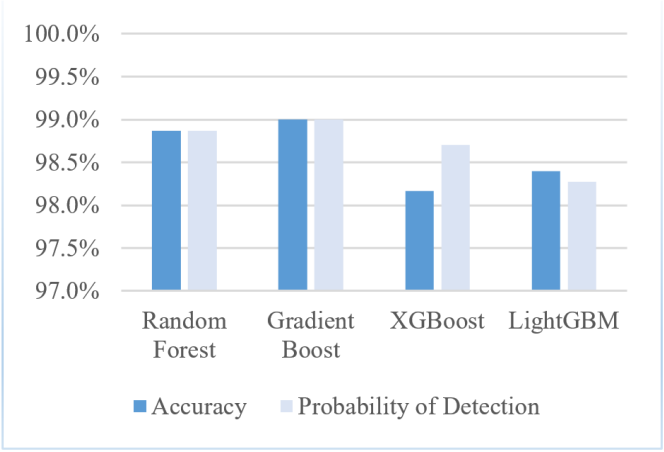

Fig. 4. Accuracy and probability of detection of the selected ML models

Nonetheless, LightGBM sustains satisfactory results in terms of the four selected metrics, where it attains a probability of detection and accuracy of more than 98% and misdetection and false alarm probabilities of less than 1.8%. While XGBoost has overall an average performance compared to other models, the probability of misdetection is high (2.4%). This will have a significant impact on the detection system since many attacks would go undetected.

Furthermore, LightGBM is considered as a lighter version of XGBoost, which might generally lead to a degradation in performance. However, we can observe that LightGBM achieves better results in terms of accuracy and misdetection probability. Although XGBoost has a slightly lower probability of false alarm than LightGBM, this will not have a major impact on jamming attacks detection since only some legitimate signals will be identified as attacks, hence it is not regarded as a crucial metric.

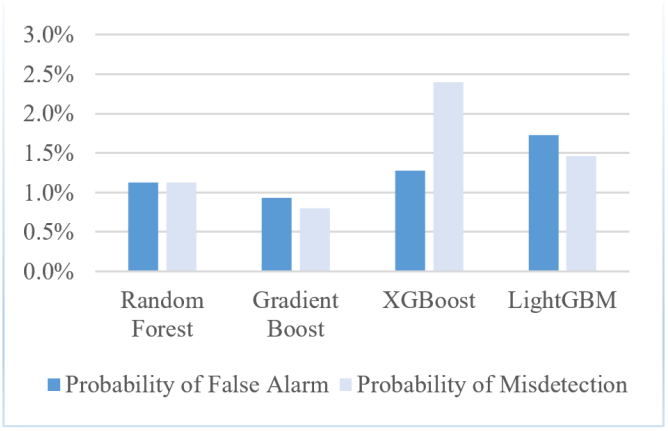

Fig. 5. Probabilities of false alarm and misdetection of the selected ML models

Since the aim of this study is to propose a fast, lightweight, and accurate technique to be implemented on the UAV, we also compared the previous ML classifiers according to the processing time, the algorithm's computational complexity, and the occupied memory for both the training and prediction. The results are shown in Table IV, where n is the number of data samples, p is the number of features, t is the number of trees, n' is the number of data samples after the GOSS algorithm, and b is the number of bundles, i.e., selected features through EFB method, where $n' < n$ and $b < p$.

From Table IV, we can perceive that LightGBM significantly outperforms the other classifiers in terms of processing time, computational complexity, and used memory in the prediction. LightGBM is 21 times faster and occupies two times less memory during the prediction than the Gradient Boost. Compared to Random Forest, LightGBM is more than 34 times faster and requires 1.4 times less memory. Moreover, its computational complexity is substantially lower than the other algorithms. In contrast, Random Forest has the worst processing time and complexity.

TABLE IV. ML Models Evaluation Results

		RF	GBM	XGB	LGBM
Processing Time (s)		1.422	0.876	0.053	0.041
Complexity		$O(nlog(n)pt)$ [21]	$O(npt)$ [22]	$O(np)$ [20]	$O(n'b)$ [20]
Memory Size (MiB)	**Train**	0.992	1.438	2.188	1.867
	Detect	0.023	0.035	0.016	0.016

These results clearly indicate that compared to the three models (XGBoost, Gradient Boost, and Random Forest), LightGMB is a more suitable model for jamming attack detection in UAVs, due to not only its high accuracy and low misdetection rate but also to its low processing time and memory usage, both of which are critical requirements in

UAVs.

IV. Conclusion and Future Work

Deceptive jamming attacks are among the most damaging threats since they are simple to carry out and difficult to detect. Existing techniques are impractical for UAV networks, as they require high computational cost, power, memory usage, or additional hardware. In this paper, we proposed a lightweight ML model, LightGBM, to detect deceptive jamming attacks on a UAV network and compared its performance with those of Random Forest, Gradient Boost and, XGBoost. We first simulated deceptive jamming attacks in a UAV network and collected the signals. Then, we used Spearman correlation and Extra Tree classifier to discard correlated features and choose the most important ones in the corresponding dataset. We also applied grid search to ensure that the ML models' hyperparameters were optimally tuned. The models' performance was evaluated in terms of detection, misdetection, accuracy, false alarm, processing time, computational complexity, and memory usage. The results show that our proposed model outperforms the other models in terms of processing time, computational complexity, and prediction memory usage. Future work will include investigating the impacts of jamming attacks on UAV networks.

Acknowledgment

The authors acknowledge the support of the National Science Foundation (NSF) through the Award Number: 2006674.

References

[1] M. R. Manesh and N. Kaabouch, "Cyber-attacks on unmanned aerial system networks: Detection, countermeasure, and future research directions," *Computers & Security*, vol. 85, pp. 386–401, 2019.

[2] N. I. Mowla, N. H. Tran, I. Doh, and K. Chae, "Federated learning-based cognitive detection of jamming attack in flying ad-hoc network," *IEEE Access*, vol. 8, pp. 4338–4350, 2019.

[3] M. P. Arthur, "Detecting signal spoofing and jamming attacks in UAV networks using a lightweight IDS," *International Conference on Computer, Information and Telecommunication Systems (CITS)*, pp. 1–5, 2019.

[4] H. Sedjelmaci, S. M. Senouci, and N. Ansari, "A hierarchical detection and response system to enhance security against lethal cyber-attacks in UAV networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 9, pp. 1594–1606, 2017.

[5] C. Greco, P. Pace, S. Basagni, and G. Fortino, "Jamming detection at the edge of drone networks using multi-layer perceptrons and decision trees," *Applied Soft Computing*, vol. 111, p. 107806, 2021.

[6] B. Duan, D. Yin, Y. Cong, H. Zhou, X. Xiang, and L. Shen, "Anti-jamming path planning for unmanned aerial vehicles with imperfect jammer information," *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 729–735, 2018.

[7] X. Gao, H. Jia, Z. Chen, G. Yuan, and S. Yang, "UAV security situation awareness method based on semantic analysis," *IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pp. 272–276, 2020.

[8] O. Puñal, I. Aktaş, C.-J. Schnelke, G. Abidin, K. Wehrle, and J. Gross, "Machine learning-based jamming detection for IEEE 802.11: Design and experimental evaluation," *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pp. 1–10, 2014.

[9] A. Marttinen, A. M. Wyglinski, and R. Jäntti, "Statistics-based jamming detection algorithm for jamming attacks against tactical MANETs," *IEEE Military Communications Conference*, pp. 501–506, 2014.

[10] Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi, and N. Kaabouch, "A novel jamming attacks detection approach based on machine learning for wireless communication," *International Conference on Information Networking (ICOIN)*, pp. 459–464, 2020.

[11] B. Upadhyaya, S. Sun, and B. Sikdar, "Machine learning-based jamming detection in wireless IOT networks," *IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, pp. 1–5, 2019.

[12] S. Gecgel, C. Goztepe, and G. K. Kurt, "Jammer detection based on artificial neural networks: A measurement study," *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*, pp. 43–48, 2019.

[13] K. Vijayakumar, P. Ganeshkumar, M. Anandaraj, K. Selvaraj, and P. Sivakumar, "Fuzzy logic–based jamming detection algorithm for cluster-based wireless sensor network," *International Journal of Communication Systems*, vol. 31, no. 10, p. e3567, 2018.

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

[15] S. Jaitly, H. Malhotra, and B. Bhushan, "Security vulnerabilities and countermeasures against jamming attacks in wireless sensor networks: A survey," *2017 International Conference on Computer, Communications and Electronics (Comptelix)*, pp. 559–564, 2017.

[16] A. A. Fadele, M. Othman, I. A. T. Hashem, I. Yaqoob, M. Imran, and M. Shoaib, "A novel countermeasure technique for reactive jamming attack in internet of things," *Multimedia Tools and Applications*, vol. 78, no. 21, pp. 29 899–29 920, 2019.

[17] D. Liu, S.-Y. Cho, D.-m. Sun, and Z.-D. Qiu, "A spearman correlation coefficient ranking for matching-score fusion on speaker recognition," *TENCON IEEE Conference*, pp. 736–741, 2010.

[18] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, no. 1, pp. 3–42, 2006.

[19] G. I. Webb and Z. Zheng, "Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 8, pp. 980–991, 2004.

[20] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, pp. 3146–3154, 2017.

[21] H. B. Li, W. Wang, H. W. Ding, and J. Dong, "Trees weighting random forest method for classifying high-dimensional noisy data," *IEEE international conference on e-business engineering*, pp. 160–163, 2010.

[22] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.