Two methods for Jamming Identification in UAV Networks using New Synthetic Dataset

Joseanne Viana [†‡], Hamed Farkhari[*†], Luis Miguel Campos [*], Pedro Sebastião [†‡],
Francisco Cercas [†‡], Luis Bernardo [§‡], Rui Dinis[§‡],
[†]ISCTE – Instituto Universitário de Lisboa, Av. das Forças Armadas, 1649-026 Lisbon, Portugal
[*]PDMFC, Rua Fradesso da Silveira, n. 4, Piso 1B, 1300-609, Lisboa, Portugal
[‡]IT – Instituto de Telecomunicações, Av. Rovisco Pais, 1, Torre Norte, Piso 10, 1049-001 Lisboa, Portugal
[§]FCT – Universidade Nova de Lisboa, Monte da Caparica, 2829-516 Caparica, Portugal;
Emails : joseanne_cristina_viana@iscte-iul.pt, Hamed_Farkhari@iscte-iul.pt, luis.campos@pdmfc.com,
pedro.sebastiao@iscte-iul.pt, francisco.cercas@iscte-iul.pt, lflb@fct.unl.pt, rdinis@fct.unl.pt

***Abstract*—Unmanned aerial vehicle (UAV) systems are vulnerable to jamming from self-interested users who utilize radio devices to disrupt UAV transmissions. The vulnerability occurs due to the open nature of air-to-ground (A2G) wireless communication networks, which may enable network-wide attacks. This paper presents two strategies to identify Jammers in UAV networks. The first strategy is based on a time series approach for anomaly detection where the available signal in the resource block is decomposed statistically to find trends, seasonality, and residues. The second is based on newly designed deep networks. The combined techniques are suitable for UAVs because the statistical model does not require heavy computation processing, but is limited to generalizing possible attacks that might occur. On the other hand, the designed deep network can classify attacks accurately, but requires more resources. The simulation considers the location and power of the jamming attacks and the UAV position related to the base station. The statistical method technique made it feasible to identify 84.38% of attacks when the attacker was at a distance of 30 m from the UAV. Furthermore, the Deep network's accuracy was approximately 99.99 % for jamming powers greater than two and jammer distances less than 200 meters.**

***Index Terms*—Cybersecurity, Convolutional Neural Networks (CNNs), Deep Learning, Jamming Detection, Jamming Identification, UAV, Unmanned Aerial Vehicles, 4G, 5G;**

I. Introduction

When it comes to the 5G communication system, the deployment of unmanned aerial vehicles (UAVs) is a game-changer. They allow faster and more flexible network services in the sky at higher data rates because they have complete control over their movement and a high probability of establishing robust line-of-sight (LoS) communication links[1], [2] and [3]. Yet, UAV transmission is vulnerable to attacks and interference because of the open nature of air-to-ground (A2G) wireless communication links and the A2G channel connections that may present opportunities for attacks in the network. As a result, it is critical and vital to identify threats and protect UAV communications[4]. In wireless communications, encryption and encoding techniques are commonly employed to ensure security by preventing unwanted access and intentional interference. Nevertheless, maintaining encryption systems requires a lot of effort and resources [5]. Consequently, encrypted UAVs' communication may not be feasible. Therefore, attack identification mechanisms become fundamental in UAV networks. Commonly used jamming detection algorithms such as packet delivery ratio and received signal strength with a missed detection rate are presented by [6]. Statistical models have lately been recognised as feasible methods for monitoring network activity in wireless communications and detecting suspicious attacks via the use of wireless channel properties rather than encryption keys. In [7], the authors propose a jamming detection method using a Naive Bayes classifier trained on a limited sample of data that considers only transmission noise effects in wireless scenarios. Cheng et al. [8] describes a Bayesian method for jamming detection. In [9], the authors offer a jamming detection strategy for GNSS-based train localization that makes use of singular value decomposition (SVD). Lu et al. [10] present a technique for detecting jamming in power networks that is both efficient and resilient. Most of the studies' computations did not take channel impacts into account. Considering machine learning and Deep Networks, Youness et al. [11] analyze the signal properties that may be used to detect jamming signals, and create a dataset based on these parameters. They utilize the random forest method, the support vector machine algorithm (SVM), and a neural network algorithm to classify the features extracted by the jamming signal. Li et al. [12] also identify jamming samples using signal-extracted features, but the author adds another way to detect attacks that utilizes 2D samples and pretrained networks (i.e. AlexNet, VGG-16, ResNet-50). Although, with pre-trained networks, we may utilize transfer learning to adapt the network to a new dataset without having to design it from scratch. Certain pre-trained networks are enormous and need significant computational processing in order to categorize information which may be unsuitable for UAVs. While embedded statistical models and deep network techniques in the cloud or at the edge can monitor and analyze channel degradation caused by jamming attacks, there is a lack of publicly available research on attack detection in UAV communications. Rather than identifying attacks, most research in this field focuses on prevention, namely anti-jamming measures and non-traditional ways to avoid jamming.

We aim to demonstrate that it is feasible to identify attacks in the receiver block of the UAV by combining a Seasonal Trend Decomposition (STL) time series analyzer with a unique deep network architecture that has much fewer layers than the well-known pretrained networks and does not rely on transfer learning techniques.

The remainder of the paper is structured as follows: First, the detailed system model is presented in Section II. It describes the dataset used, the statistical approach, and the suggested deep network architecture. The assessment of both approaches is summarized in Section III. Finally, section IV concludes this paper.

Notations: Scalar variables are denoted by lower-case letters (a, b,...), vectors are denoted by boldface lower-case letters **(a, c,...)**, and matrices are denoted by boldface capitals **(A, B,...)**. Lower case letters denote time-domain variables, whereas upper case letters indicate frequency-domain variables.

A. *Contributions and Motivation*

With regard to jamming detection and the associated challenges utilizing deep networks and statistical methods, there is a lack of public research and accessible data. Taking this into consideration, the following highlights some of the contributions made by this paper:

- A general case study model that takes into account the jamming power and distances between the jamming attacker and the base station in relation to the authenticated UAV that uses average received signal power in the resource block, Signal-To-Noise-Ratio (SNR), average-noise, average-transmitted-power, path-loss, and shadowing.
- A statistical model for jamming detection using data from the UAV's reception resource block.
- A simpler Convolutional Neural Network-Long Short Term Memory (CNN-LSTM) architecture for jamming detection.
- Simulation results for both of the presented techniques.
- A comparison of two jamming detection technique performances in terms of accuracy over attacker distance and power.

Additionally, we offer a table representation of the confusion matrix for both the training and test data sets. We devise a strategy that increases performance while using the fewest CNN layers.

II. System Model

We analyze a UAV jamming scenario in which a communication link exists between the base station and the UAV, referred to as an air-to-ground (A2G) connection and there are jamming attackers in unknown locations on the ground or in the air that can deliberately jam the signal received by the authenticated UAV. Although, we use the Single Carrier (SC) transmission scheme, the jamming detection algorithms are applicable to any transmission technique. We investigate the reception power in the authenticated UAV using two distinct approaches: one of which relies on time series statistical models and the other on deep networks. Each component is explained as follows:

A. *dataset*

The data set simulates the received signal in the UAV resource block considering slow fading effects in the transmission channel, specifically (pathloss and shadowing). The frequency domain channel $H_{(k,d)}$ is represented as in 1,

$$H_{(k,d)} = \frac{\sum_{i=1}^{N_{rays}} \alpha_i(\tau) + \exp(-j2f\pi\tau)}{\sqrt{(PLS)}} \tag{1}$$

For $H_{(k,d)}$, f is the frequency band, α is the attenuation of the multipath ray, and τ is the propagation delay. We adopt the Rician model to describe the multipath rays. The path loss is estimated using the UAV and base station locations $p_{id,t} = [x_{uav,t}, y_{uav,t}, z_{uav,t}]^T$, $[x_b,\, y_b,\, z_b]^T$ (in meters) and the 3D Euclidean distance equation $||p_{bs} - p_{uav}||^2$ respectively. The reference point is at $d = 10m$ and S the shadowing is a random variable modeled as $S|_{db} \sim \mathcal{N}(0,\, \sigma^2)$. The received power $Y_{(k,d)}$ is calculated using 2,

$$Y_{(k,d)} = H_{(k,d)} X_k + N, \tag{2}$$

X_k is the frequency domain representation of the transmitted signal x_k^R, and $N \sim \mathcal{N}(\mu,\, \sigma_k^2)$ is the noise in the channel, while k is an available frequency in the bandwidth.

The jamming signal takes into account the same properties of the reliable signal considering path loss. In the experiment, the jammer focuses on F_k « F frequencies inside the bandwidth B available for transmission with the gain $(P/P_J)I$, where I is the percentage of the slot occupied by the jammer. The jammer is more powerful than the signal in the majority of the dataset samples (i.e. $P_J > P_S$). The formula in 3 shows how the total noise is affected by the jamming power received, where N_k is the noise without interference.

$$E[|N_{k,Tot}|^2] = \frac{F}{F_{kJ}} \frac{P_j}{P_S} E[|N_k|^2], \tag{3}$$

The dataset contains 483,540 transmission blocks or samples Sa, with N steps each $\{Sa_k; k = 0, 1, ..., N-1\}$ where N is the FFT size in the frequency domain. The received signal is then classified according to the following categories: Good-Normal, Bad-Normal, Good-Jamming, and Bad-Jamming. "Normal" defines a non-jamming signal while good and bad channels are distinguished by $SNR = 20$ and $SNR = 1$, respectively. Additionally, we vary the jammer and base station positions as well as the jammer power in the experiment. Fig 1 depicts two jamming samples available in the dataset. The top illustrates a jamming sample in a good channel. The bottom shows a jamming signal in a bad channel. The dataset contains only the received power categorized in the four classes previously mentioned.

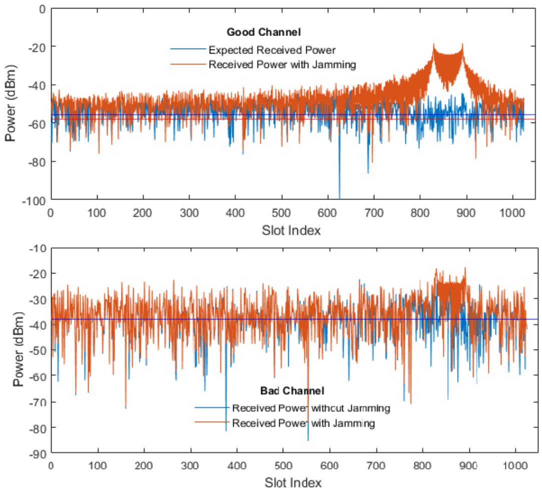


Fig. 1: Received signal with jamming in Good and Bad Channels.

Table I compares the mean received power differences at the UAV when the signal is jammed and when the signal is not jammed with the same power as the authenticated signal at a distance of 30m from the UAV. The base station is about the same distance from the UAV. According to the table, the jammer modifies the mean power in the resource block, depending on the jammer's power and position relative to the base station and the authorized UAV.

TABLE I: Mean Difference Between Channels with and Without Jamming

Mean Power difference (dBm)	No Jamming	Jamming
Good Channel	0	4.189
Bad Channel	0	0.592

B. Statistical methods

The statistical model chosen was STL [13]. It takes into account the decomposition of the signal into trends, seasonal, and residuals. Figs 2 and 3 illustrate a representative sample of both jammed and unjammed decomposed signals. The top chart in both figures shows a combination of three samples from the dataset in a sequence. In fig 2, the jamming power of the attacker is five times greater than the signal received from the base station. The jamming location is 30m away from the UAV, while the UAV placement is 90m away from the base station. In fig 3, there is no jamming attacker and the base station location is identical to that in fig 2.

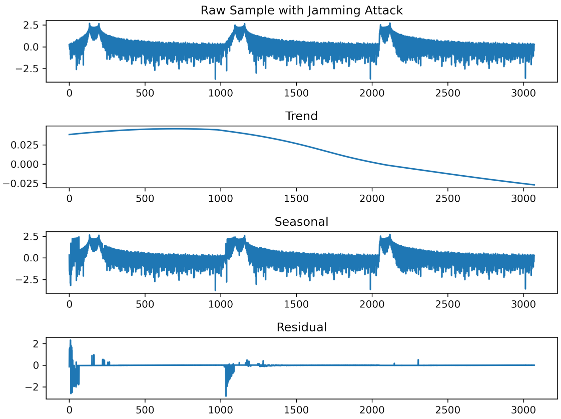


Fig. 2: STL decomposition of a normalized sample in the present of jamming attacker.

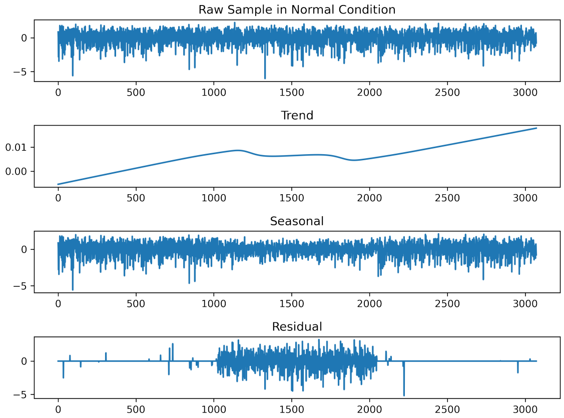


Fig. 3: STL decomposition of a normalized sample without jamming signal.

STL is an acronym for "Seasonal and Trend decomposition based on locally weighted regression (Loess)". This is a technique for decomposing time series data into trend, seasonal, and residual components. After removing the current trend estimate, the seasonal component of the cyclic sub-series is calculated using seasonal smoothing. Next, the predicted seasonal component is smoothed using lowpass smoothing. Finally, the deseasonalized series is smoothed once more with trend smoothing in order to provide an estimate of the trend component. This procedure is repeated numerous times in order to improve the component estimates' accuracy[13].

In the experiment, each sample contains elements that degrade communication, such as noise and slow fading components. The channel and the noise effects are independent of each other. We assume that the jamming effect will manifest itself in multiple samples in the case of a jamming attacker. Then, we exploit the jamming attack's periodicity to identify it using STL. We assume the jamming attack is applied in a certain narrow channel bandwidth from 8% to 10% . After the third resource block reception, first, we combine three samples in a sequence. Second, we apply the STL decomposition and then reconstruct the signal again using 4. Finally, we calculate the sample's error as in 5. If there is a pattern in the sample, the number of errors is smaller; consequently, the sample is classified as a jamming signal. In a normal situation without a jamming effect, there is no specific repeated pattern, and we see more errors in the STL decomposition reconstructed signal. In order to classify the signals correctly, we use root mean square error (RMSE) for binary classification to determine the presence or absence of jamming effects in the experiment as in 6. Lastly, we apply Support Vector Machine (SVM), Logistic Regression, and Random Forest algorithms to split the features in the classes. The difference between the dataset described in the previous section and the one used in the STL is the concatenation of three resource blocks in one sample. While developing the experiment, we noticed it is fundamental to accurately define the $period$ in order to get good results from STL.

$$S_a = T + S + R \tag{4}$$

$$Error = S_a - S_r \tag{5}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i}^{N} Error_i^2} \tag{6}$$

In 4, 5 and 6, S_a and S_r are the original and reconstructed samples, and T, S, and R stand for trend, seasonal, and residual components, respectively. N is the length of a sample and we use $RMSE$ as a feature for binary classification.

C. Convolutional Neural Networks

In the experiment using the convolutional neural network (CNN) joined with Long short-term memory (LSTM), we developed an architecture capable of achieving 99 % accuracy with a small number of CNN layers, number of filters, and kernel sizes in the convolutional layers. Our CNN architecture uses three convolutional layers: one LSTM, one drop-out, a fully connected layer, and the output layer for classification as Fig 4 illustrates. Max-pooling is a common layer used in

CNNs, but it might result in the loss of critical information in certain topologies. According to Geofrey Hinton "pooling is a mistake" and for these reasons, we replace the pooling layer with strides in our convolutional layers. Authors in [14] and [15] reported that using very small weight decay (L2 regularization) values such as 5x10^{-4}, and 4x10^{-5} in convolutional layers are critical for performance purposes and they should be precisely chosen. After executing the grid search algorithm, we found the an optimal L2 regularization and used it for all CNN layers in the experiment.

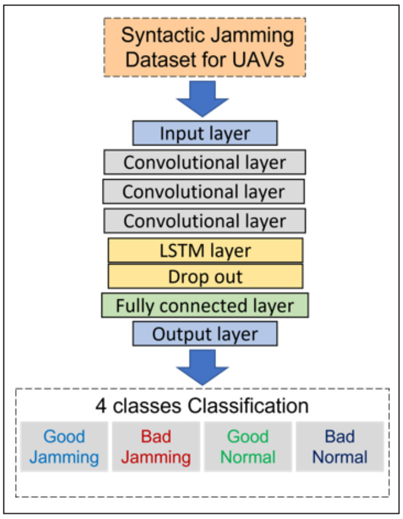


Fig. 4: Proposed CNN-LSTM Architecture.

For the deep network, we use a single sample dataset. Initially, we partition the dataset into 70% for training and 30% for testing. In the first phase, we divide the training section into two sub-sections: training and validation. We then apply the grid search algorithm to determine the deep network's hyperparameters. The hyperparameters are as follows: the number of CNN filters, the kernel sizes, the strides, the batch size, the learning rate, the number of regularization terms, and the drop out percentage. In the second phase, we employ a 5-fold cross validation procedure during the training phase to ensure accuracy and avoid overfitting.

III. Experimental Results

The results of the suggested algorithms are detailed below. First, we look at the statistical data for jamming detection. Then we'll look into deep networks for classification, loss, and accuracy in training and testing. The CNN model is trained and tested in a system with a Nvidia RTX 3090 GPU. The jamming attacker signal power ratio ranges between one and twenty. The distances between the UAV and the base station, and the UAV and jamming attacker varies between 10 and 350 meters. The shadowing variance was adjusted to 4 [16].

A. Statistical model

Fig 5 depicts the RMSE between original signal and reconstructed one after STL decomposition in the BoxPlot. The diagram shows that both distributions can be split using binary classification. We merge three resource block into a sequence where each of them is the size of $N = 1024$ in length, and we use N as a period parameter for the STL decomposition algorithm. After calculating the reconstructed signal, we use RMSE as a feature to detect the jamming attack with respect to distance and power. The overall accuracy of this method for all scenarios using the three different classifiers is about 70%, as is shown in Fig 6, and varies according to the jamming power ratio and distance of the UAV from the base station and the attacker. In some cases, depending on the jammer and base station location relative to the UAV, the accuracy can increase up to 84.38% by employing an SVM classifier that outperforms the other two classifiers.

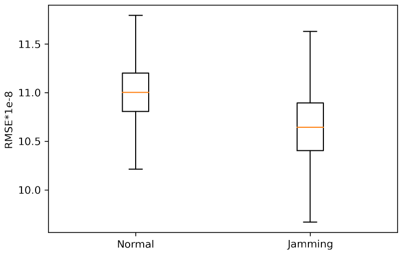


Fig. 5: Boxplot of RMSE between original signal and reconstructed signal for two classes.

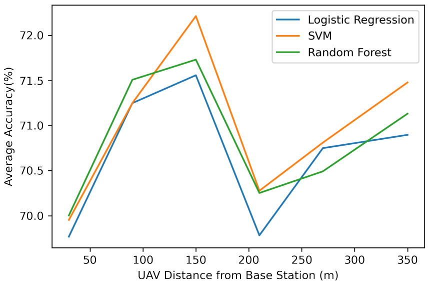
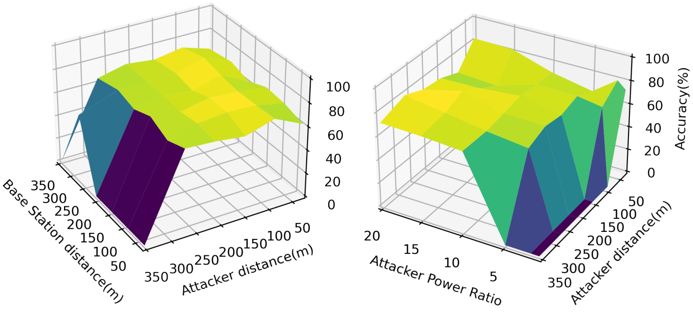


Fig. 6: Overall statistical average accuracy for different attacker powers and distances per fixed UAV distance from base station.

Fig 7 (a) and (b) illustrate the $accuracies$ of the STL model at various attacker distances and power ratios using the SVM classifier. In (a), the accuracy decreases with increasing jammer distance, i.e. when the jammer is 350m away, the accuracy in the statistical model is reduced. When the jamming power P_j decreases as specified in (b), it is difficult for the algorithm to differentiate low-power jammers and prominent channel effects such as fading, path loss, and shadowing. Due to the statistical model's low computational requirements, it may readily be implemented in UAVs for user packet transmission and Command and Control (C2) links.



(a) (b)

Fig. 7: (a) Accuracy for $P_j = 5P_s$. (b) Accuracy for $Bs_d = 350m$.

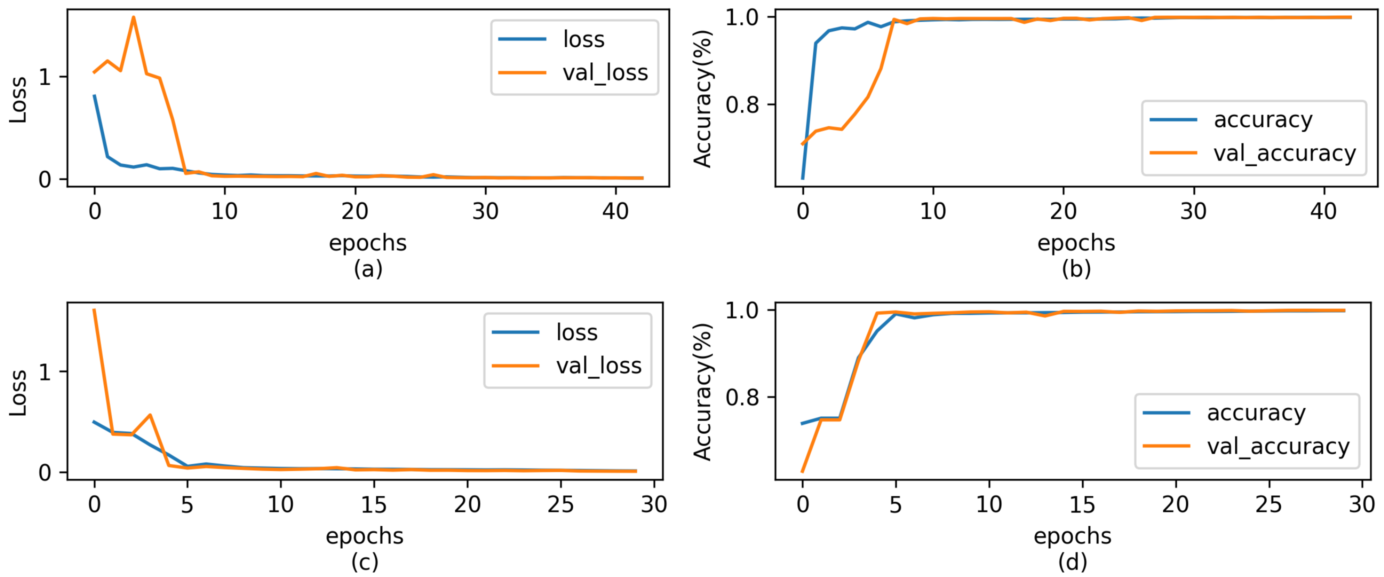


Fig. 8: Convergence of deep network during training with 4 steps. a) loss of model 1. b) *accuracy* of model 1. c) loss of model 2. d) *accuracy* of model 2.

B. Convolutional Neural Networks

In the CNN simulation, the slot size is set to N=1024. The only parameter the experiment uses for jamming detection is the signal received in the resource block. During the CNN performance configuration phase, we notice that adding layers is preferable to increasing the number of filters in each layer. Additionally, the regularization component of the CNN and the fully connected layer are crucial for achieving performance improvements since, in both cases, the parameters demand appropriate adjustment. The LSTM layer is added to take advantage of the sequence memory characteristics in order to increase robustness. Also, the filter numbers and the kernel sizes are implied in the overall trainable parameters. We achieve the same performance by employing two fully connected layers of 50 nodes each rather than a single layer of 100 nodes. These adjustments result in a decrease in the total number of trainable parameters from around 100k to 53k. Table II presents the hyperparameters of our deep network.

TABLE II: Deep Network Configuration Parameters.

Deep network Parameters	Value
base learning rate	3.16x10^{-3}
base batch size	32
conv-1 filters, kernel size, strides	4, 8, 4
conv-2 filters, kernel size, strides	4, 4, 2
conv-3 filters, kernel size, strides	4, 3, 1
LSTM	100
drop-out	0.4
dense	100
softmax	4

We use L2 regularization terms equal to 1x10^{-6} and 1x10^{-5} in the convolutional layers and in the fully connected layer for both kernels and biases, respectively. The initial batch size is 32 for the deep network. Then the grid search algorithm defines the learning rate as 3.16x10^{-3}. After that, we increase the learning rate and batch size to 0.2, and 2048, respectively. The new batch size and learning rate increases GPU (RTX 3090 with 24GB Ram) use from 30% to 92% of the limit of processing capacity. Consequently, the batch size is limited to 2048. Following that, we train our deep network in the different steps for validation accuracy. At each training step, if the performance declines compared with the previous step, the training process is immediately stopped, the previous model weights are loaded, and the training process at that step is repeated with a new lower learning rate and batch size. These steps are used to achieve 80, 90, 95, and 99.99% validation accuracy, and as the training process progresses through each step, we save the model and continue the training process. By employing this strategy, we minimize the overshooting effect in the deep network and shorten the overall training time for five models in 5-fold cross validation. As an example, fig 8 shows the convergence of two models from 5 models in cross validation. It shows 99.99% *accuracy* for all the five models in 5-fold cross validation with a maximum of 40 epochs.

TABLE III: Confusion matrix of 4 classes classification for test data by CNN-LSTM network.

	Good Normal	Bad Normal	Good Jamming	Bad Jamming
Good-Normal	36281	0	0	0
Bad-Normal	0	36219	0	62
Good-Jamming	0	0	36281	0
Bad-Jamming	0	385	0	35896

Table III shows the confusion matrix the test set using the CNN-LSTM algorithm. We obtain the correct classification for all samples with good channels using the designed deep network and there is minimal misclassification in the case of bad channels. Specifically, we have 62 misclassifications in the absence of jamming and 385 when jamming is present which represents less than 1% of the total samples analyses. Table

TABLE IV: The result of test set for 4-Classes classification by proposed deep network.

	precision	recall	f1-score	support
Good-Normal	1.00	1.00	1.00	36281
Bad-Normal	0.99	1.00	0.99	36281
Good-Jamming	1.00	1.00	1.00	36281
Bad-Jamming	1.00	0.99	0.99	36281

IV provides more specific details of the precision and f-score parameters in the experiment.

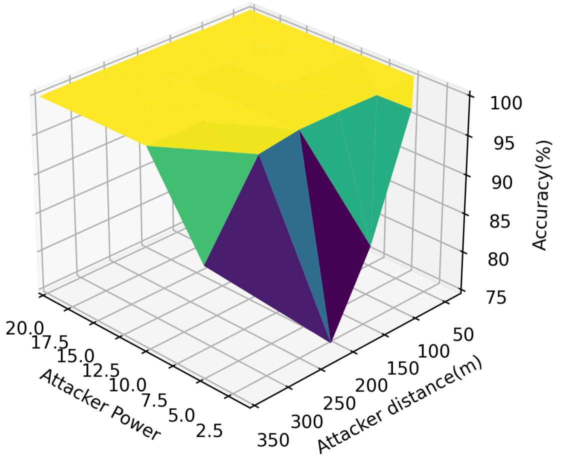


Fig. 9: CNN Accuracy for base station distance 30m.

One of the following setup techniques may be used to reduce the total number of trainable parameters. First, three CNN layers with eight filters each, followed by a LSTM and two fully connected layers with fifty nodes each. Alternatively, three CNN layers with four filters, followed by a LSTM and only one fully connected layer with one hundred nodes can be used. We discover that the second one converges more quickly and with fewer epochs.

Fig 9 depicts the CNN model's performance related to the $accuracy$ over a range of attacker distances and power ratios. CNN may struggle to identify low-power jammers depending on the channel situation, the same as in the STL statistical model, but in all other circumstances, CNN obtains 99.99% correct classifications.

IV. Conclusion

This article offered a solution composed of two techniques for identifying jamming attacks in UAV networks. The first one is based on a time series method for detecting patterns using the STL decomposition technique. The second is based on convolutional neural networks. The signal analysed by both approaches relied on the resource blocks received by the UAV. Using the time series analysis, it was possible to identify 84.38% of the attacks when the $SINR$ of the jamming signal was high and the UAV was closer to the attacker than to the base station. While using the deep networks accuracy was 99.99% in the jamming cases and false alarms occurred in less than 1% of the cases. The combined method is appropriate for UAVs since the statistical model is restricted in its ability to classify all conceivable attacks. However, the deep network can classify all attacks, but requires additional resources.

Acknowledgment

This research received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie Project Number 813391"

References

[1] Zeeshan Kaleem et al. "UAV-empowered disaster-resilient edge architecture for delay-sensitive communication". In: *arXiv* December (2018), pp. 124–132. DOI: 10.1109/MNET.2019.1800431.

[2] Syed Ahsan Raza Naqvi et al. "Drone-Aided Communication as a Key Enabler for 5G and Resilient Public Safety Networks". In: *IEEE Communications Magazine* 56.1 (2018), pp. 36–42. DOI: 10.1109/MCOM.2017.1700451.

[3] Fei Qi et al. "UAV Network and IoT in the Sky for Future Smart Cities". In: *IEEE Network* 33.2 (2019), pp. 96–101. DOI: 10.1109/MNET.2019.1800250.

[4] Hoon Lee et al. "UAV-Aided Secure Communications With Cooperative Jamming". In: *IEEE Transactions on Vehicular Technology* 67.10 (2018), pp. 9385–9392. DOI: 10.1109/TVT.2018.2853723.

[5] Liang Xiao et al. "User-Centric View of Unmanned Aerial Vehicle Transmission Against Smart Attacks". In: *IEEE Transactions on Vehicular Technology* 67.4 (2018), pp. 3420–3430. DOI: 10.1109/TVT.2017.2785414.

[6] Aleksi Marttinen, Alexander M. Wyglinski, and Riku Jäntti. "Statistics-Based Jamming Detection Algorithm for Jamming Attacks against Tactical MANETs". In: *2014 IEEE Military Communications Conference*. 2014, pp. 501–506. DOI: 10.1109/MILCOM.2014.90.

[7] Yuxin Shi et al. "Efficient Jamming Identification in Wireless Communication: Using Small Sample Data Driven Naive Bayes Classifier". In: *IEEE Wireless Communications Letters* 10.7 (2021), pp. 1375–1379. DOI: 10.1109/LWC.2021.3064843.

[8] Maggie Cheng, Yi Ling, and Wei Biao Wu. "Time Series Analysis for Jamming Attack Detection in Wireless Networks". In: *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*. 2017, pp. 1–7. DOI: 10.1109/GLOCOM.2017.8254000.

[9] Jian-Cong Li et al. "Jamming Identification for GNSS-based Train Localization based on Singular Value Decomposition". In: *2021 IEEE Intelligent Vehicles Symposium (IV)*. 2021, pp. 905–912. DOI: 10.1109/IV48863.2021.9575412.

[10] Zhuo Lu, Wenye Wang, and Cliff Wang. "Modeling, Evaluation and Detection of Jamming Attacks in Time-Critical Wireless Applications". In: *IEEE Transactions on Mobile Computing* 13.8 (2014), pp. 1746–1759. DOI: 10.1109/TMC.2013.146.

[11] Youness Arjoune et al. "A Novel Jamming Attacks Detection Approach Based on Machine Learning for Wireless Communication". In: *2020 International Conference on Information Networking (ICOIN)*. 2020, pp. 459–464. DOI: 10.1109/ICOIN48656.2020.9016462.

[12] Yuchen Li et al. "Jamming Detection and Classification in OFDM-Based UAVs via Feature- and Spectrogram-Tailored Machine Learning". In: *IEEE Access* 10 (2022), pp. 16859–16870. DOI: 10.1109/ACCESS.2022.3150020.

[13] Robert B. Cleveland et al. "STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion)". In: *Journal of Official Statistics* 6 (1990), pp. 3–73.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[15] François Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions". In: *CoRR* abs/1610.02357 (2016). arXiv: 1610.02357. URL: http://arxiv.org/abs/1610.02357.

[16] Walid Saad et al. *Wireless Communications and Networking for Unmanned Aerial Vehicles*. Cambridge University Press, 2020. DOI: 10.1017/9781108691017.