RFI and Jamming Detection in Antenna Arrays with an LSTM Autoencoder

Christos Ntemkas and Antonios Argyriou
Department of Electrical and Computer Engineering, University of Thessaly, 38334 Volos, Greece.

***Abstract*—Radio frequency interference (RFI) and malicious jammers are a significant problem in our wireless world. Detecting RFI or jamming is typically performed with model-based statistical detection or AI-empowered algorithms that use an input baseband data or time-frequency representations like spectrograms. In this work we depart from the previous approaches and we leverage data in antenna array systems. We use Fourier imaging to localize spatially the sources and then deploy a deep LSTM autoencoder that detects RFI and jamming as anomalies. Our results for different power levels of the RFI/jamming sources, and the signal of interest, reveal that our detector offers high performance without needing any pre-existing knowledge regarding the RFI or jamming signal.**

***Index Terms*—RFI, Interference, RFI detection, Antenna Arrays, Beamforming, Jamming, Autoencoder, AI, LSTM, Deep Neural Network.**

I. Introduction

Radio frequency interference (RFI) and malicious RF jamming are significant problems for a plethora of applications. RF sensing systems must detect and classify these types of signals so that appropriate counter-measures can be implemented. Detection can be realized with model-based algorithms that incorporate knowledge regarding the undesired signal that must be detected, e.g., cyclo-stationarity for RFI that consists of communication signals [1]. Since RFI is typically assumed to originate from wireless communication, modulation classification with Artificial Intelligence (AI) techniques [2], [3], can be re-purposed for RFI detection. By using features like spectrograms [3], [4], higher order moments [2], etc., AI can distinguish the unique behavior of RFI originating from wireless modulated signals and detect it. However, not all RFI consists of communication signals, or even signals of known behavior for which we can collect data and train AI models. Classifying these non-canonical signals where the injectors of RFI were malicious jammers was first considered in [5], and later in [4], [6]–[8]. Some of these systems may even exploit hardware impairments at the jammers for detection [9], [10]. In this work we show that for RFI or jammers to be detected, and potentially classified, we need to introduce data from additional sources, namely multiple antennas.

Our approach is pragmatic today due to the move in higher communication frequencies in modern communication systems (5G/6G). These systems use multiple antennas that are deployed to form array structures. Antenna arrays are used in radar, radio astronomy, and wireless communication, among other applications. Fig. 1 illustrates a 2D uniform rectangular array (URA). The purpose of an antenna array is to sample spatially an incoming RF wavefront. It can also be steered electronically to a desired spatial direction [11], something that can be accomplished with beamforming (BF) algorithms. The objective of BF is to maximize the output power of the array for signals incoming from a desired angle-of-arrival (AoA). This behavior is captured visually in the so-called *array response* plot that can be 2D or 1D. A 1D cut of such a 2D array response can be seen in Fig. 2. The first beam in this figure indicates that the signal originating from the desired *look angle* of 0^o will have the maximum power while signals from other AoAs will be attenuated.

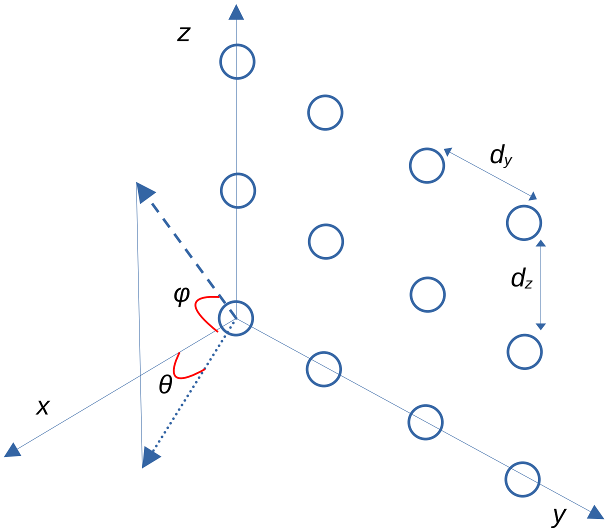


Fig. 1: URA geometry with the circles indicating the positions of the antennas in the yz plane. The incoming signal arrives from direction θ, ϕ.

In Fig. 2 signals from all other directions are considered as RFI or malicious jamming. Hence, we desire for them to be suppressed. Some beamforming techniques like the minimum variance distortionless response (MVDR), or Capon beamformer [12], have RFI suppression capabilities since they create nulls in the spatial direction of the RFI. But before RFI/jamming suppression schemes are deployed we must detect these events.

In this work, we propose an RFI and jamming detection scheme that uses spatially-sampled data from an array that operates in passive receiving mode. We use these data to calculate the correlation matrix of the time-domain samples across the antennas and then calculate its discrete Fourier

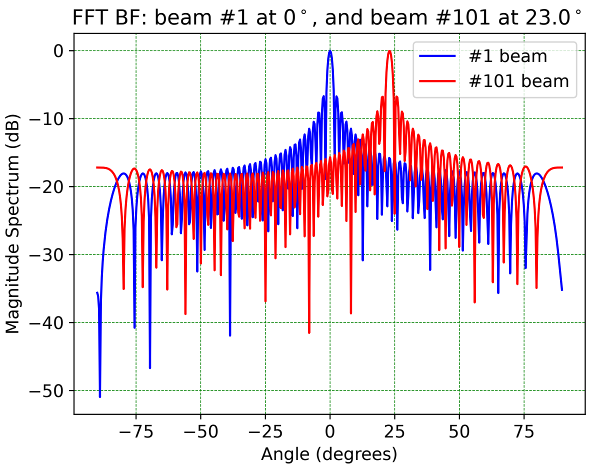


Fig. 2: Two beams obtained with DFT beamforming. We set $\phi = 0$ and vary θ.

transform (DFT). As we will show next, when we have a 2D URA, the 2D-DFT of this correlation matrix provides an estimate of the signal power as a function of azimuth θ and elevation ϕ. The result is an image of the sources present in the received signal versus their spatial location [13]. These images are used for training an AI system, namely a deep neural network (DNN), which in our case is selected to be a Long Short-Term Memory (LSTM) autoencoder. Besides the images, an additional feature we provide as input to the DNN is *the desired look angle*. In the presence of RFI or jamming, the 2D-DFT images will contain peaks in spatial directions beyond the look angle. These are anomalies for which the autoencoder has not been trained and they can be detected.

II. Data Model & Assumptions

URA Model: We consider a redundant array of antennas, and in particular a URA with a number of $N\times M$ elements (Fig. 1). The unit vector oriented towards the directions of azimuth θ, elevation ϕ in Cartesian coordinates is $\mathbf{r_1}(\theta,\phi) = \cos(\phi)\cos(\theta)\mathbf{x_1} + \cos(\phi)\sin(\theta)\mathbf{y_1} + \sin(\phi)\mathbf{z_1}$, where $[\mathbf{x_1}\ \mathbf{y_1}\ \mathbf{z_1}]^T$ is the Cartesian unit vector. Let us assume that the antennas are deployed along the y, z dimensions, and are spaced apart by d_y and d_z meters, respectively. So its x coordinate is zero. The location of the n, m-th element of the array is given by the Cartesian coordinates $\mathbf{r} = [nd_y\ \ md_z]$, i.e., the Cartesian position vector is $\mathbf{r}_{n,m} = nd_y\mathbf{y_1} + md_z\mathbf{z_1}$. For an incoming wave from direction ϕ, θ that impinges on the array, the phase difference between the 0,0-th and the m, n-th element depends on the *equivalent* distance $\mathbf{r}_{n,m}\cdot\mathbf{r_1}(\theta,\phi)$. For the n, m-th element we have:

$$\mathbf{r}_{n,m}\cdot\mathbf{r_1}(\theta,\phi) = nd_y\cos(\phi)\sin(\theta) + md_z\sin(\phi) \tag{1}$$

To calculate the impact of this geometry on the array data model, we have to discuss briefly some well-known but necessary aspects of the sensing instrument. Let the $MN\times 1$ vector $\mathbf{y}$ denote a single time domain sample of the baseband radio signal received at all the array elements after down-conversion to baseband and analog-to-digital conversion (ADC) [14]. When this signal is wideband with the help of a filterbank it is split into multiple narrowband frequency channels and the respective filtered baseband signals in each of these are then denoted as $\mathbf{y}(f)$. This is necessary so that the narrowband assumption holds and the effect of the geometrical delay across two elements of the array is captured only by a phase shift [14]. Then the phase shift between two elements is simply the angular wavenumber $2\pi/\lambda$ multiplied by their equivalent distance given in (1). Hence, the phase shift of the n, m-th element relative to the 0,0 is:

$$\begin{aligned} a_{n,m}(f,\phi,\theta) &= \exp\left(j\frac{2\pi}{\lambda}(\mathbf{r}_{n,m}\cdot\mathbf{r_1})\right) \\ &= \exp(j\frac{2\pi}{\lambda}(nd_y\cos(\phi)\sin(\theta)+md_z\sin(\phi))) \\ &0\le m\le M-1,\ \ 0\le n\le N-1 \end{aligned} \tag{2}$$

In the above expression, the spatial frequencies are $\cos(\phi)\sin(\theta)/\lambda$ and $\sin(\phi)/\lambda$ respectively. With the narrowband assumption in mind, we can define the steering vector (array manifold) that captures the phase difference between the received signal at each element of the 2D array. It is the $MN\times 1$ vector $\mathbf{a}$ which is frequency dependent:

$$\begin{aligned} \mathbf{a}^T = \Big[1\quad &\exp\left(j\frac{2\pi}{\lambda}(1d_y\cos(\phi)\sin(\theta)+1d_z\sin(\phi))\right)\ \ldots \\ &\exp\left(j\frac{2\pi}{\lambda}((N-1)d_y\cos(\phi)\sin(\theta)+(M-1)d_z\sin(\phi))\right)\Big] \end{aligned} \tag{3}$$

Data Model for the URA: The random signal of interest (SOI) when sampled is $x[l]$ and *is spatially narrow at the angles* ϕ, θ. $\sigma_x^2(\phi,\theta) = \mathbb{E}[|x[l]|^2]$ is the power of the SOI in the direction ϕ, θ. $z[l]$ is the sample of the overall RFI/jamming signal that also incorporates the steering vector (we do not model it like the SOI since it may come from multiple AoAs and is of no interest to process it). The received signal for the l-th time instant and the n, m-th element is:

$$y_n[l] = a_{n,m}(f,\phi,\theta)x[l] + z[l] + w[l]. \tag{4}$$

In the above $w[l]$ denotes the Gaussian noise sample. The data model for the complete URA is again an $MN\times 1$ vector:

$$\mathbf{y}[l] = \mathbf{a}x[l] + \mathbf{z}[l] + \mathbf{w}[l]. \tag{5}$$

Correlation: In theory the correlation matrix is:

$$\mathbf{R_y} = \mathbb{E}[\mathbf{yy}^H] = \mathbf{aa}^H\sigma_x^2(\theta,\phi) + \mathbf{R}_\mathbf{z}^2 + \mathbf{R}_\mathbf{w}^2. \tag{6}$$

The dimensions of this matrix are $NM\times NM$ since it contains the correlation between a single element and all the remaining ones (including itself). It is important to understand precisely the contents of this matrix for designing later our detector. Since we have a redundant array, this means that multiple antenna elements receive a signal coherently (the same phase) and so they can add them (coherent combining). This allows us to reach a simplified expression for each element of this matrix. Focusing on two antenna elements with coordinates n_1, m_1 and n_2, m_2, we set $n_1 - n_2$=k, $m_1 - m_2$=ℓ to capture the relative difference in the indexes of two elements. This difference is what matters for the calculation of correlation. Correlating the signal in (4) across these two antennas:

$$\mathbb{E}[y_{n_1,m_1}[l]y^*_{n_2,m_2}[l]] = \\ = \sigma_x^2(\theta,\phi)e^{j\frac{2\pi}{\lambda}((m_1-m_2)d_y\cos(\phi)\sin(\theta)+(n_1-n_2)d_z\sin(\phi))} + \sigma_w^2$$

If we add the the previous result across redundant antenna pairs (that satisfy $n_1 - n_2$=k, and $m_1 - m_2$=ℓ):

$$r_{\ell,k} = \sum_{\substack{\forall n_1-n_2=k\\ \forall m_1-m_2=\ell}} e^{j\frac{2\pi}{\lambda}(\ell d_y\cos(\phi)\sin(\theta)+kd_z\sin(\phi))}\sigma_x^2(\theta,\phi) \\ + \sigma_w^2, \ell\in[0,M],\ k\in[0,N]. \tag{7}$$

The last expression provides the theoretical values for each element of matrix (6). In practice we can only estimate $\mathbf{R_y}$: From the baseband data in (5) that we collect for each snapshot, we estimate the sample correlation matrix $\hat{\mathbf{R}}_\mathbf{y}$ as:

$$\hat{\mathbf{R}}_\mathbf{y} = \frac{1}{S}\sum_{s=1}^{S}\mathbf{y}_s\mathbf{y}_s^{\mathrm{H}}, \tag{8}$$

with S being the number of collected snapshots of $\mathbf{y}$. So we essentially estimate each element of this matrix as $\hat{r}_{\ell,k}$.

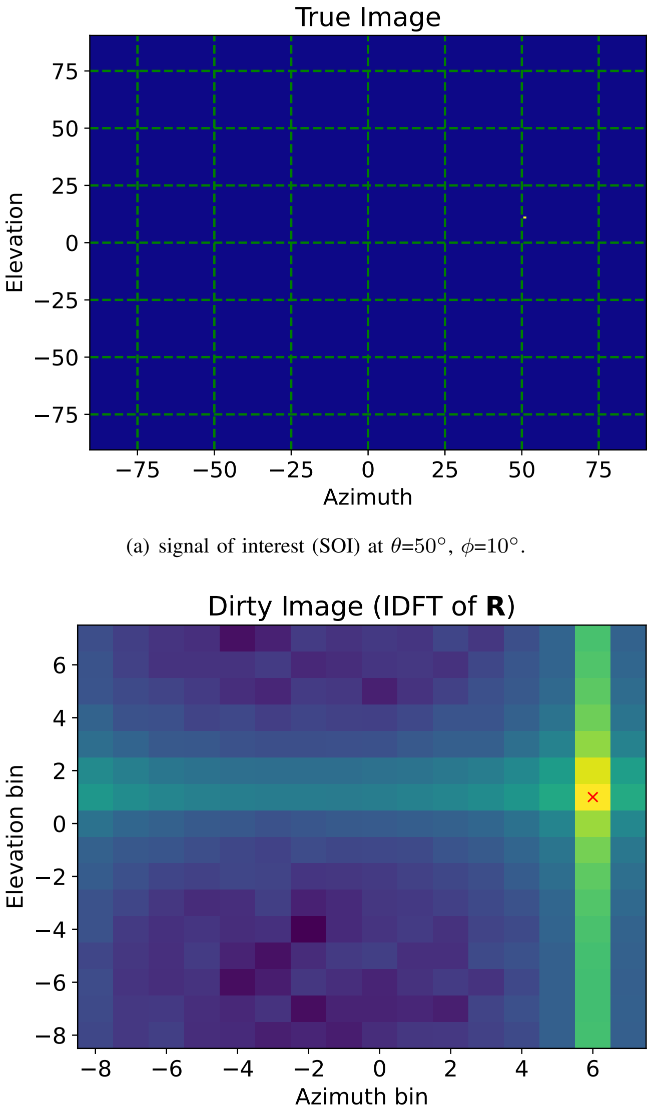
Note that our first goal is to estimate $\sigma_x^2(\theta,\phi)$ for every angle θ,ϕ (which is what constitutes our images), by using $\hat{r}_{\ell,k}$ and (7). An efficient way to do this is with the DFT. Data Pre-Processing with 2D-DFT

III. Data Pre-Processing with 2D-DFT

Imaging with data from array samples uses the correlated data. The 2D-DFT of the correlations in (7) provides the signal power over the different spatial directions [15], [16]. First we need to calculate the theoretical continuous Fourier Transform (FT) of the discrete data, namely the discrete-time FT (DTFT) [17]. The 2D-DTFT of the 2D data $r_{\ell,k}$ is:

$$G(f_1,f_2) = \sum_k\sum_\ell r_{\ell,k}e^{-j2\pi\ell\frac{f_1}{f_{s_1}}}e^{-j2\pi k\frac{f_2}{f_{s_2}}} \tag{9}$$

where f_1, f_2 are the frequencies and f_{s_1}, f_{s_2} are the corresponding sampling frequencies [18]. In our application the data $r_{\ell,k}$ are sampled spatially. Consequently, the frequencies and sampling rates have to correspond to the data sampled in the spatial domain. Our objective is to estimate the signal power from the estimated matrix $\hat{\mathbf{R}}$. Hence, we need to apply the DTFT by taking into account the signal model in (7).



(a) signal of interest (SOI) at θ=50°, ϕ=10°.

(b) 2D-IDFT of the covariance matrix.

Fig. 3: (a) Real source location, (b) input image to the DNN which also includes the location of the look angle (the red x).

By simple inspection of (7) we notice that if we want the result of the DTFT of the data $r_{\ell,k}$ in (9) to correspond to the signal power $\sigma_x^2(\theta,\phi)$, we must replace the frequencies f_1, f_2 in (9) with the spatial frequencies $\cos(\phi)\sin(\theta)/\lambda$ and $\sin(\phi)/\lambda$ respectively, and the sampling rates f_{s_1}, f_{s_2} with the spatial sampling rates $1/d_y$ and $1/d_z$ respectively. Taking then the 2D-DTFT over all the pairs of antennas we have that the estimated power of the signal over the continuous variables that indicate spatial direction ϕ, θ is:

$$\hat{\sigma}_x^2(\theta,\phi) = \sum_{k,\ell}\frac{\hat{r}_{\ell,k}}{NM}e^{-j2\pi\ell\frac{\cos(\phi)\sin(\theta)/\lambda}{1/d_y}}e^{-j2\pi k\frac{\sin(\phi)/\lambda}{1/d_z}} \\ \ell\in[0,M],\ k\in[0,N]. \tag{10}$$

Remark 1: This expression highlights the fact that with FT we are essentially doing beamforming of the correlations $r_{\ell,k}$ by setting as the BF coefficients the DTFT weights: In (10) we combine the correlation data coherently by correcting their phase offset due to the instrument geometry.

2D-DFT: Now we want to discritize the 2D-DTFT in (10) so that we can calculate it in practice. We must derive the 2D-DFT with a number of $uFFT$, $vFFT$ points that are selected

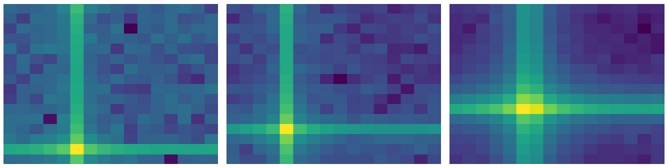


Fig. 4: 2D-DFT images with a moving jammer.

depending on the desired resolution. We index these DFT points with $u \in [\frac{-uFFT}{2}, \frac{uFFT}{2}-1], v \in [\frac{-vFFT}{2}, \frac{vFFT}{2}-1]$. The 2D-DFT of a set of 2D discrete data $r_{\ell,k}$ is then:

$$I[u,v] = \sum_{k=0}^{N-1}\sum_{\ell=0}^{M-1} r_{\ell,k} e^{-j2\pi\ell\frac{u}{uFFT}} e^{-j2\pi k\frac{v}{vFFT}} \tag{11}$$

Remark 2: A representative image of the 2D-DFT after applying (11) can be seen in Fig. 3(b) where a single point source is present. However, we must not forget that this DFT processing produces the *dirty image* with the DTFT [15]. The term dirty image is used when the result is affected by the available DFT bins, cannot capture precisely a frequency component present in the signal, resulting in sidelobes.

Conversion to Angular Bearings: One final step has to do with the interpretation of these images. We convert the result of 2D-DFT beamforming in (11) to angles θ, ϕ instead of using the binning of the 2D-DFT transform as seen in Fig. 3(b). For the DTFT to DFT conversion, we can see that the mapping from the continuous frequencies in (9) to the discrete in (11) is accomplished by setting $\frac{f_1}{f_{s_1}} = \frac{u}{uFFT}, \frac{f_2}{f_{s_2}} = \frac{v}{vFFT}$. The normalized spatial frequency over the sampling rate is between $1/2 \leq f_1/f_{s_1} \leq 1/2$. We have already substituted in (10) the spatial frequencies f_1, f_2 that are $\cos(\phi)\sin(\theta)/\lambda$ and $\sin(\phi)/\lambda$ respectively. Hence, from (10), (11) we simply solve the following:

$$\begin{aligned}\frac{f_2}{f_{s_2}} &= \frac{v}{vFFT} \Rightarrow \frac{\sin(\phi)/\lambda}{1/d_z} = \frac{v}{vFFT}\\ \frac{f_1}{f_{s_1}} &= \frac{u}{uFFT} \Rightarrow \frac{\cos(\phi)\sin(\theta)/\lambda}{1/d_y} = \frac{u}{uFFT}\end{aligned}$$

By solving the last two, the final angle locations are obtained. For the **elevation bin** v the angle is

$$\phi = \arcsin(\frac{v}{nFFT}\frac{\lambda}{d_z}), \tag{12}$$

and for the **azimuth bin** u the angle depends on the elevation:

$$\theta = \arcsin(\frac{u}{uFFT}\frac{\lambda}{d_y}\frac{1}{\cos(\phi)}) \text{ rad}. \tag{13}$$

IV. Dataset

The dataset was synthetically generated to simulate the URA responses for different SOI locations. It consists of the 2D-DFT images $I[u,v]$ (calculated in (11)). A typical image in the dataset is presented in Fig. 3(b). For producing the training dataset that contained only the SOI we varied its spatial location to cover the entire $M \times N$ 2D-DFT image. The SNR of the SOI varied from $-9\,\mathrm{dB}$ to $20\,\mathrm{dB}$ with a step size of $1\,\mathrm{dB}$. This approach generated $M \times N \times 30$ images. The training and validation datasets consisted of different perturbations of images in the time domain. The testing dataset consisted of clean images but also images that contain RFI and jammers. RFI and jamming sources can be *transient* and appear in a single image, or *moving* and *static* that appear in more than one image. Fig. 4 has an example of a moving jammer. The testing dataset consisted of RFI/jamming sources with an interference to noise ratio (INR) ranging from 0 to 30 dB.

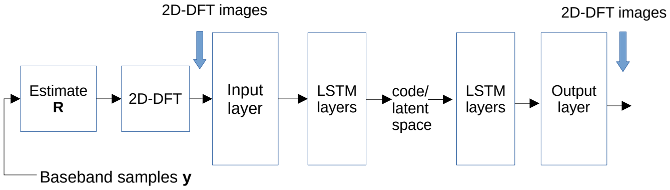


Fig. 5: LSTM autoencoder with 2D-DFT image inputs.

V. Sparsity-Inducing LSTM Autoencoder

The neural network (NN) model used in this work is an LSTM autoencoder. The autoencoder compresses the input data into a condensed representation, which it then reconstructs it to its original form (Fig. 5). We will now write formally the operations of the auto-encoder for the 2D-DFT of matrix $\mathbf{R}$ that was calculated from our spatially-sampled data. The input vector denoted as $\mathbf{i}$, consists of the data given by (11). Its reconstruction denoted as $\hat{\mathbf{i}}$. The output of the encoder is the latent representation $\mathbf{h}$ which is also referred to as the *code* [19]. $\mathbf{W}_E, \mathbf{W}_D$ are the parameters of the encoder and decoder respectively that are trained, and $\mathcal{L}$ is the loss function that is the MSE plus the sparsity loss:

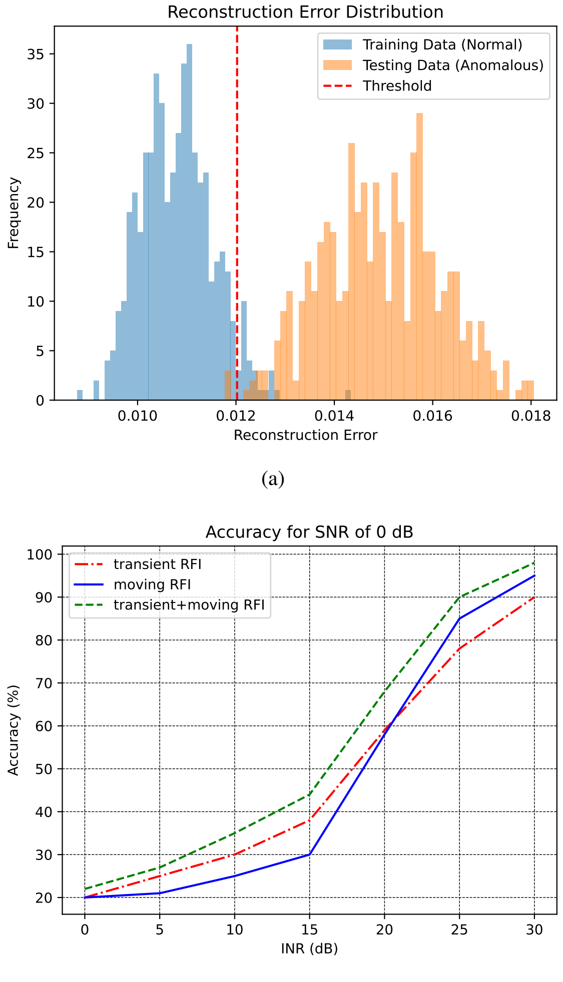
$$\begin{aligned}&\mathbf{h} = f_{\mathrm{E}}(\mathbf{i}, \mathbf{W}_E), \quad \hat{\mathbf{i}} = f_{\mathrm{D}}(\mathbf{h}, \mathbf{W}_D),\\ &\mathcal{L} = \frac{1}{T}\sum_t \Big(\|\mathbf{i}^{(t)} - \hat{\mathbf{i}}^{(t)}\|_2^2 + \alpha\|\mathbf{h}^{(t)}\|_1\Big).\end{aligned} \tag{14}$$

The number of training batches is T, $\|\cdot\|_2$ is the L2 norm, α is the hyperparameter that determines the weight of the sparsity-inducing L1 norm [19]. Contrary to image classification with a convolutional NN (CNN), our input observation consists of a sequence of P successive images of size $M \times N$. In this work, the input layer has as input $M \times N$=128 $\times$128 features, and P=10. Hence, we detect RFI/jamming based on events occurring in a sequence of P images and not a single image.

The NN architecture implemented in this case was comprised of 3 LSTM layers at the encoder f_{E}, and 3 at the decoder f_{D}. The input/output dimensions of the three LSTM layers are 128x128/64x64, 64x64/32x32, 32x32/16x16, i.e., the number of nodes in the code layer (code size) is 16x16. The resulting LSTM autoencoder model has $\approx$ 23M parameters. The Adam optimizer was used as the solver during training with a batch size of 32 observations. We fine-tuned the learning rate to 0.02 and the sparsity weight to $\alpha = 0.001$.

VI. Results

After training, the model is used for anomaly detection [20]. Detection of RFI or jamming is declared when, during testing, the reconstruction error of the input sequence of P images is more than a threshold (more than the 95th percentile of the training reconstruction error). In Fig. 6(a) we show the distribution of the reconstruction error of the autoencoder for an INR of 20dB and a very low SOI power with an SNR of 0 dB. The anomalous data are easily distinguishable even for this low SOI power. In Fig. 6(b) we present the detection accuracy for different values of the RFI INR. We also explore different types of RFI/jamming sources. In the low INR regime, the RFI source(s) are very weak and so their presence cannot be easily detected. As the INR is increased, we can detect them better despite the low power SOI. Notice that the most challenging case is the transient RFI/jammer. Note also that as moving jammers are added, we have a performance increase due to the image being more distinguishable even for low INR. Due to lack of space, we do not present a SNR higher than 0 dB since it is more favorable for our system as expected.



(a)

(b)

Fig. 6: (a) Distribution of the error after image reconstruction with the auto-encoder. (b) Accuracy vs. INR for different number and types of RFI sources in the testing dataset.

VII. Conclusions

In this paper, we presented a new method to detect RFI or jamming with DNNs but using as input features the spatially-sampled antenna array data that are converted to images. This is contrary to spectrograms and time-domain baseband samples that are used as input in existing RFI and jamming classification schemes. Our performance evaluation suggests that this data modality can lead to high classification accuracy of RFI or jamming when we have antenna array systems.

References

[1] W. A. Gardner, "The spectral correlation theory of cyclostationary time-series," *Signal Processing*, vol. 11, no. 1, pp. 13–36, Jul. 1986.

[2] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over the Air Deep Learning Based Radio Signal Classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, Feb. 2018.

[3] D. Varkatzas and A. Argyriou, "Waveform Manipulation Against DNN-based Modulation Classification Attacks," in *MILCOM 2023 - 2023 IEEE Military Communications Conference (MILCOM)*. Boston, MA, USA: IEEE, Oct. 2023, pp. 580–585.

[4] J. Yu, M. Alhassoun, and R. M. Buehrer, "Interference Classification Using Deep Neural Networks," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, Nov. 2020, pp. 1–6.

[5] D. Karagiannis and A. Argyriou, "Jamming attack detection in a pair of RF communicating vehicles using unsupervised machine learning," *Vehicular Communications*, vol. 13, pp. 56–63, Jul. 2018.

[6] Y. Ghanney and W. Ajib, "Radio Frequency Interference Detection using Deep Learning," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, May 2020, pp. 1–5.

[7] T. Oyedare, V. K. Shah, D. J. Jakubisin, and J. H. Reed, "Keep It Simple: CNN Model Complexity Studies for Interference Classification Tasks," in *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, May 2023, pp. 1–6.

[8] B. Singh, V. P. Singh, A. Bajpai, C. Bulla, S. Parikh, and R. Maranan, "Unsupervised Learning for Detection of Radio Frequency Interference Signals," in *2024 International Conference on Intelligent Systems for Cybersecurity (ISCS)*, May 2024, pp. 1–6.

[9] A. Argyriou, "False Target Detection in OFDM-based Joint RADAR-Communication Systems," in *2023 IEEE Radar Conference (RadarConf23)*, May 2023, pp. 1–6.

[10] M. S. Arif, S. Muhaidat, A. Argyriou, and P. C. Sofotasios, "Performance of AI-Empowered Anti-Jamming Communications under Hardware Impairments," in *2024 IEEE Middle East Conference on Communications and Networking (MECOM)*. Abu Dhabi, United Arab Emirates: IEEE, Nov. 2024, pp. 274–279.

[11] H. L. Van Trees, *Optimum Array Processing*. New York: Wiley-Interscience, 2002.

[12] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proceedings of the IEEE*, vol. 57, no. 8, pp. 1408–1418, Aug. 1969.

[13] M. A. Richards, *Fundamentals of Radar Signal Processing*, 3rd ed. McGraw-Hill Education, 2022.

[14] A.-J. Van Der Veen, S. J. Wijnholds, and A. M. Sardarabadi, "Signal Processing for Radio Astronomy," in *Handbook of Signal Processing Systems*, S. S. Bhattacharyya, E. F. Deprettere, R. Leupers, and J. Takala, Eds. Cham: Springer International Publishing, 2019, pp. 311–360.

[15] A. R. Thompson, J. M. Moran, and G. W. Swenson, *Interferometry and Synthesis in Radio Astronomy*, ser. Astronomy and Astrophysics Library. Cham: Springer International Publishing, 2017.

[16] B. F. Burke, F. Graham-Smith, and P. N. Wilkinson, *An Introduction to Radio Astronomy*, 4th ed. Cambridge: Cambridge University Press, 2019.

[17] C. L. Byrne, "Signal Processing: A Mathematical Approach, Second Edition," 2015.

[18] A. V. Oppenheim and R. W. Schafer, Eds., *Discrete-Time Signal Pro-*

cessing, 3rd ed., ser. Prentice Hall Signal Processing Series. Upper Saddle River Munich: Pearson, 2010.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, ser. Adaptive Computation and Machine Learning. Cambridge, Mass: The MIT press, 2016.

[20] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, “Long Short Term Memory Networks for Anomaly Detection in Time Series,” *Computational Intelligence*, 2015.