Abstract—Conventional anti-jamming methods mainly focus on preventing single jammer attacks with an invariant jamming policy or jamming attacks from multiple jammers with similar jamming policies. These anti-jamming methods are ineffective against a single jammer following several different jamming policies or multiple jammers with distinct policies. Therefore, this article proposes an anti-jamming method that can adapt its policy to the current jamming attack. Moreover, for the multiple jammers scenario, an anti-jamming method that estimates the future occupied channels using the jammers’ occupied channels in previous time slots is proposed. In both single and multiple jammers scenarios, the interaction between the users and jammers is modeled using recurrent neural networks (RNNs). The performance of the proposed anti-jamming methods is evaluated by calculating the users’ successful transmission rate (STR) and ergodic rate (ER), and compared to a baseline based on deep Q-learning (DQL). Simulation results show that for the single jammer scenario, all the considered jamming policies are perfectly detected and a high STR and ER are maintained. Moreover, when 70% of the spectrum is under jamming attacks from multiple jammers, the proposed method achieves an STR and ER greater than 75% and 80%, respectively. These values reach 90% when 30% of the spectrum is under jamming attacks. In addition, the proposed anti-jamming methods significantly outperform the DQL method for all the considered jamming scenarios.

Index Terms—Jamming recognition, multiple jammers, recurrent neural network (RNN).

I. INTRODUCTION

Wireless communication networks are susceptible to jamming attacks due to their shared and open nature. Jamming attacks cause performance degradation or denial of service by disrupting communication links in wireless networks. Therefore, it is necessary to adopt anti-jamming methods to mitigate such attacks. Various anti-jamming techniques have been proposed in the literature; however, the majority of available techniques focus on preventing a single type of jamming attack. For instance, Xuan et al. [2], Nan et al. [3], Pourranjbar et al. [4], D’Oro et al. [5], and Pourranjbar et al. [6] mainly focused on mitigating reactive jammer attacks while Slimeni et al. [7] and Yao and Jia [8] performed anti-jamming against sweeping jammers. Some machine-learning-based anti-jamming techniques that can work against several jammer types, such as [9], assume that during interactions between legitimate users and jammers, the policies of the jammers remain unchanged. Here, if the policy of a jammer changes, legitimate nodes must be retrained, resulting in a performance loss. Therefore, there is a pressing need for an anti-jamming method capable of mitigating multitype jammer attacks.

When confronted with a single jammer that can adopt different jamming policies, it is necessary to first recognize the jammer’s policy and then select an appropriate countermeasure [1]. Jamming recognition techniques are mainly studied in the context of radar [10], [11], [12], [13], with a focus on detecting the jammer’s type utilizing the jamming signals. In [10], the power spectrum of the jamming signal is used to determine the type of jammer. Shao et al. [11] utilized the fast Fourier transform (FFT) of the jamming signal to recognize the jamming policy while in [12] and [13], the time-domain signal is used. Cai et al. [14] employed convolutional neural networks (CNNs) to recognize the jamming type using a waterfall plot of the spectrum.

From a practical perspective, implementing the recognition techniques from [10], [11], [12], and [13] requires accurate samples from the jamming signal. Moreover, the recognition technique in [14] is limited to simple jamming policies and requires a large data set for training. Thus, it is necessary to develop an anti-jamming technique that can work with a small data set for training and can be extended to different types of jamming.

The presence of multiple jammers with different jamming policies gives rise to additional challenges in wireless networks. However, most previous works, such as [2], [3], [4], [5], [7], and [8], consider the single jammer case. Some works, such as [15], [16], [17], and [18] consider multiple jammers with similar jamming policies. Specifically, Garnaev et al. [15], [16] formulated the interaction between a user and several jammers as a noncooperative game where the utility functions of the jammers are the same. In [15], the latency
is considered as the utility function of the jammers while in [16] the SINR is set as the jammers’ utility function. Van Huyhn et al. [17] proposed to harvest the reactive jammers’ energy and use backscatter to relay the user’s data. Similarly, the work in [18] considers a multifunction wireless system and employs a backscatter as a redundant communication path when the main path is under threat from reactive jammers. Moreover, the works in [19], [20], and [21] focus on the localization of the multiple jammers and do not propose any anti-jamming methods. In the context of unmanned aerial vehicles (UAVs), the works in [22], [23], and [24] attempt to determine the trajectory for UAVs that are under attack by multiple jammers to avoid jamming attacks. Gingras et al. [25] and Lundén et al. [26] proposed a channel allocation method based on spectrum sensing. In this context, the users share their local spectrum sensing with each other to select the channels that are less likely to be jammed.

Although the works in [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], and [26] study scenarios with multiple jammers, they suffer from several drawbacks, which make them impractical against multiple jammers with different jamming policies. The assumption of similar policies at the jammers is a major drawback in all the above mentioned works. Moreover, some of these works, such as [15] and [16], assume the availability of the channel gains between the user and jammers, which is not practical. The considered schemes in [17] and [18] are restricted to specific system models because a backscatter is employed to relay the users’ data in case of jamming attacks. The proposed solutions in [22], [23], and [24] can only be applied to UAVs, and, therefore, is unsuitable for a wider range of applications. Gingras et al. [25] and Lundén et al. [26] proposed to monitor the jammers’ occupied channels in previous time slots to allocate free channels to the users in future time slots. This approach is practical against multiple jammers with multiple jamming policies since they focus on determining free channels in future time slots instead of selecting a countermeasure for a specific type of jamming attack. However, the exploitation of the obtained knowledge is inefficient since these methods only utilize the information of the spectrum in the last time slot while the information of the spectrum occupancy in previous time slots can be utilized to better understand the spectrum occupancy pattern.

In light of the above discussion, it is obvious that, for wireless networks, the problem of jamming attacks from multiple jammers with multiple jamming policies needs to be investigated. Thus, in the second part of this article, we propose an anti-jamming technique for multiple jammers with different jamming policies. The main contribution of this article is to propose two novel anti-jamming methods against a single jammer and multiple jammers who are capable of launching attacks with different jamming policies. In order to defend against the single jammer, we develop an anti-jamming method that requires a small data set for training and is practical against several jamming types. In this context, the jammer’s type is first detected, and then an appropriate countermeasure is selected. For multiple jammers with different jamming policies, we propose to estimate the jammers’ future behavior from their occupied channels in previous time slots. In both proposed anti-jamming techniques, the occupied channels of the jammers in previous time slots are stored, and used to estimate the jamming policy or jammers’ future occupied channels. Due to the sequential nature of the interaction between users and jammers, we propose to use recurrent neural networks (RNNs) in both cases. To evaluate our methods, we calculate the ergodic rate (ER) and perform different simulations considering different jamming policies. Our results show that in the single jammer scenario, the jamming policies are detected perfectly with high accuracy within a short period, and as a result, the best countermeasure is employed, leading to a high successful transmission rate (STR) and ER. Moreover, against multiple jammers, the proposed anti-jamming technique achieves an STR and ER higher than 75% when 70% of the spectrum is jammed. This value rises to 90% when 30% of the spectrum is jammed.

The remainder of this article is organized as follows. Section II presents the system model and problem formulation. In Sections III and IV, we present the proposed anti-jamming methods for the single and multiple jammer scenarios, respectively. Simulation results are shown in Section V and conclusions are drawn in Section VI.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Model

We consider a wireless network composed of an access point (AP), N users, and J jammers. We study two distinct scenarios based on the number of jammers in the network. In the first scenario (SC1), each user is located next to a jammer capable of attacking with various jamming policies, while in the second scenario (SC2), we assume that each user is attacked by multiple jammers with different jamming policies. In both scenarios, the AP, users, and jammers are randomly distributed in the network, as shown in Fig. 1. In both scenarios, we assume that the network is constantly under jamming attacks. In SC2, each user is assumed to be located within the communication range of at least one jammer. We assume that all channels follow a Nakagami fading model while the path loss is evaluated as $(\tau/\tau_0)^{-\psi}$, where $\tau$, $\tau_0$, and $\psi$ denote the distance...
between the nodes, the reference distance, and the attenuation factor, respectively. Time is divided into time slots, where at each time-slot, each user and the AP communicate with each other through a channel selected from \( L \) channels. The packet transmitted by a user over a frequency channel is successfully received when that channel is not jammed or interfered by other users. Four types of jamming policies are considered, where in each time slot, each jammer can jam several frequency channels. The considered jamming policies are as follows.

1) Random jammer, which chooses its channels randomly.
2) Sweeping jammer, which periodically shifts its full energy over multiple frequency channels.
3) Reactive jammer, which listens to channels repeatedly and jams channels after a single time slot from sensing an activity.
4) Combat jammer, which chooses a number of channels randomly and jams them for a number of consecutive time slots.

In SC1, the jammer selects a policy from the above-mentioned policies while in SC2, we assume that each user is attacked by a group of jammers employing all of the above-mentioned jamming policies. In both scenarios, we assume that each user can sense the jamming signals in the frequency channels to detect occupancy. In this context, if the sensed amplitude of the base-band signal of a channel is higher than a specific threshold, the channel is counted as occupied, otherwise, it is assumed to be free.

### B. Problem Formulation

We now formally formulate the interaction between users and the jammers as an optimization problem. To this end, assuming that the vector of the indices of all frequency channels allocated to users under policy \( \pi \) is given by

\[
\mathbf{c}^\pi = [c_1^\pi, c_2^\pi, \ldots, c_L^\pi]
\]

where \( c_k^\pi \) denotes the index of the \( k \)th user’s selected channel under policy \( \pi \), the instantaneous sum rate of the network at time slot \( t \) is defined as

\[
R = \sum_{k=1}^{N} \mathbb{I}(c_k^\pi) \log_2 \left( 1 + \frac{\Omega_k |h^\pi_{kk}|^2}{\delta^2} \right)
\]

where \( h^\pi_{kk} \) is the channel gain between the \( k \)th user and the AP in frequency channel \( c_k \), which includes both large-scale and small-scale fading, \( \Omega_k \) is the \( k \)th user’s power, \( \delta^2 \) is the noise power, and \( \mathbb{I}(\cdot) \) is an indicator function given by

\[
\mathbb{I}(c_k^\pi) = \begin{cases} 
1, & \text{if } c_k^\pi \text{ is jamming and collision free} \\
0, & \text{otherwise}.
\end{cases}
\]

Our goal is to obtain a channel allocation strategy \( \pi^* \) for the users’ channel allocation can be formulated as the following optimization problem:

\[
\begin{align*}
\max_{\pi} & \quad R \\
\text{s.t.} & \quad |u_i| \leq 1 \ \forall i \in J \text{ and } |\mathbf{c}| = N \ \forall t
\end{align*}
\]

where \( u_i \) is a vector containing the indices of the \( i \)th jammer’s selected frequencies. The constraints \(|u_i| \geq 1\) and \(|\mathbf{c}| = N\) imply that each jammer jams at least one channel and one frequency channel is assigned to each user at each time slot, respectively.

In each iteration of the interaction between the users and the jammers, the users must find the optimal channels. Problem (4) is a discrete optimization problem aiming to maximize the users’ sum rate by selecting the optimal channels between the users and the AP. In order to maximize the users’ sum rate, the rate of each user can be maximized since

\[
\max \left( \sum_{k=1}^{N} \mathbb{I}(c_k^\pi) \log_2 \left( 1 + \frac{\Omega_k |h^\pi_{kk}|^2}{\delta^2} \right) \right)
\]

holds. As a result, in order to obtain the maximum sum rate of the network, it is necessary to obtain the channel allocation that maximizes each user’s rate. In this context, the optimal choice for each user is to select the channel that has the highest channel gain between the user and the AP among free channels.

Given that the channel gains are random variables, the maximum achievable rate of each user is different from one channel realization to another. Consequently, we take the average of the rate or ER. In order to characterize the maximum achievable ER, we consider an ideal case each user knows the jammed channels in the upcoming time slot. We use a Nakagami fading distribution with average channel power gain \( \lambda \) to model the channel gain between each user and the AP. The total number of jammed channels around the \( k \)th user is \( u_k \), the ER of the \( k \)th user is obtained using Proposition 1.

**Proposition 1:** The ER of the \( k \)th user when the highest channel gain between the user and the AP is selected

\[
R_{Ek} = \frac{\Omega_k}{\delta^2 \ln(2)} \int_0^\infty \frac{1}{1 + \frac{\lambda_k}{\delta^2}} \left( \frac{\gamma(\text{in}, \text{out})}{\Gamma(\gamma, \text{out})} \right)^{(L-u_k)} dx
\]

where \( \gamma(\cdot, \cdot) \) is the lower incomplete gamma and \( \Gamma(\cdot, \cdot, \cdot) \) is the upper incomplete gamma function, respectively.

**Proof:** See Appendix A.

**Proof:** When the users interfere with each other, the ER of each user can be obtained by slightly modifying (6). In this context, the number of free channels for each user reduces to \( L - N - u_k + 1 \). Thus, the ER of the \( k \)th user is

\[
R_{Ek} = \frac{\Omega_k}{\delta^2 \ln(2)} \int_0^\infty \frac{1}{1 + \frac{\lambda_k}{\delta^2}} \left( \frac{\gamma(\text{in}, \text{out})}{\Gamma(\gamma, \text{out})} \right)^{(L-N-u_k)} dx
\]

In some cases, it is recommended to select the users’ channels following a random policy that can minimize the probability

\[\text{Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.} \]
of being traced by the jammers. In this context, the ER of the $k$th user can be obtained using Proposition 2.

**Proposition 2:** The ER of the $k$th user when the user selects its channels randomly is given as

$$R_{ek} = \frac{1}{\Gamma(m)\ln(2)}G_{2,1}^{3,1}(\frac{\delta^2m}{\Omega_y^2}, 0, 0, m)$$

where the term $G$ denotes the Meijer G-function.

**Proof:** See Appendix B.

The equation in (8) is valid for both the ideal and interference scenarios. For all the aforementioned scenarios, the ergodic sum rate can be obtained by taking the sum of all the users’ rates. We note that the obtained ER using (6) and (8) are the maximum achievable ERs since they are calculated assuming that each user knows the other users’ and jammers’ occupied channels in future time slots.

In order to solve (4), the jammers’ selected channels at the next time slot, i.e., $A_{t+1}$, must be known; however, in realistic scenarios, this information is not available. Therefore, next, we propose RNN-based anti-jamming methods to predict the occupied channels by the jammers in the next time slot.

### III. Proposed Anti-Jamming Method Against Single-Jammer

We now introduce the proposed anti-jamming method in SC1. Since the user is attacked by a jammer that can jam with different jamming policies, we propose to first recognize the jamming policy, and then select an appropriate countermeasure against the jammer. In this context, the interaction between the user and the jammer is sequential. We, thus, employ RNNs because of their ability to process data with a sequential nature. RNNs are artificial neural networks that can learn patterns and long-term relationships from time series and sequential data.

At each time slot, an RNN takes the previous hidden state and the input, and generates the updated hidden state and output based on the conditions of the input vector, $d_t$, and the input, $y_t$.

$$d_t = \sigma(K_d d_{t-1} + V_d y_t + g_d)$$

$$y_t = \sigma(K_y d_t + g_y)$$

where $\omega$ is the input, $d_t$ is the hidden state at time slot $t$, $\sigma$ is an activation function, $V_d$ is the weight of the input vector, $g_d$ and $g_y$ are the bias terms, and $K_d$ and $K_y$ denote the weights of the hidden layer and the output, respectively. High-depth and recurrent connections in conventional RNNs cause a vanishing gradient problem. This vanishing gradient challenge is addressed in gated recurrent unit (GRU) [28] models by controlling the inputs using multiple gates in a hidden layer.

In GRUs, the controller gate $r_t$ is set to one or zero to update the hidden layer based on the conditions of the input and previous hidden layer. The overall process is shown in Fig. 2. At time slot $t$, the GRU states are updated as follows:

$$r_t = S(W_r \omega_t + U_r d_{t-1} + g_r)$$

$$z_t = S(W_z \omega_t + U_z d_{t-1} + g_z)$$

$$\hat{d}_t = \tanh(W_d \omega_t + U_d (z_t \odot d_{t-1}) + g_d)$$

$$d_t = (1 - r_t) \odot d_{t-1} + r_t \odot \hat{d}_t$$

where $W_r$, $U_r$, $W_z$, $U_z$, $W_d$, $U_d$, $g_r$, $g_z$, and $g_d$ are learning parameter matrices and vectors. Here, the operator $\odot$ represents the Hadamard product, and $S$ is the Sigmoid function.

A feedforward layer (FL) with an output size equal to the number of considered jamming classes is considered after GRU units. Then, the output of the FL is passed through a SoftMax layer to generate a probability distribution vector for the jamming type classes. It is noted that the predicted class is obtained by the index of the output array that has the highest probability value. In Fig. 3, we show the proposed network and in Table I, the structure of the proposed network is detailed.

We divide the proposed recognition technique into training and testing. In the training phase, which is an offline process,
TABLE I
STRUCTURE OF THE DEEP LEARNING MODELS

| Proposed RNN models  | Single jammer | Multiple jammers |
|----------------------|---------------|-----------------|
| Input-size           | $b \times 2L$ | $a \times 2L$   |
| Hidden-size          | 64            | 64              |
| Number of layers     | 4             | 4               |
| Feed-forward size    | 64 x C        | 64 x L          |
| Activation function  | Softmax       | Sigmoid         |
| Output-size          | C             | L               |

The interaction between the user and the jammer is simulated. In contrast, the testing is an online process that takes place during the interaction between the user and the jammer.

The interaction between the user and the jammer is simulated for $T$ consecutive time slots. During the simulation, a channel is randomly assigned to the user and the jammer’s responses for all of the considered jamming policies, i.e., random, sweeping, reactive, and combat, are simulated and the channels of the users, the jammer, and corresponding jamming domains, sweeping, reactive, and combat, are simulated and the responses for all of the considered jamming policies, i.e., randomized.

The collected data from the simulations is then used to train the proposed RNN. Here, the elements with column indices from 1 to $2L$ are the inputs while elements in the $2L + 1$th column are the corresponding class targets. During every step of the training, a matrix consisting of $a$ consecutive vectors is fed to the RNN and a vector with the same number of elements as the number of classes $C$ is generated by the network. We employ the cross-entropy as the loss function to train the network.

In the testing phase, similar to the training process, the user’s and jammer’s occupied channels in the last $b$ time slots are saved. In this context, the selected channels by the user and jammer during the last $b$ time slots are given to the trained network, and the network then generates a vector. The class of the jammer is determined by the index of the highest value in the output vector. Anti-jamming can be easily selected once the jamming policy and corresponding number of jammed channels are identified. In this context, the future behavior of the combat, sweeping, and reactive jammers can be predicted from the detected class of the jammer and the jammer’s selected channels in previous time slots. Thus, the user can find the best channel that maximizes its rate among free channels. For instance, if the trained network detects that the channels are jammed by a sweeping jammer which jams three channels per time slot, users predict that the sweeping jammer will jam channels with indices 5–7 in the coming time slot when the indices of jammed channels in previous time slot are 2–4. Thus, users will select channels other than channels with indices 5–7. For random jammers, the future channels that the jammer will select cannot be determined. In this case, the user selects a channel randomly among all channels. The proposed algorithm is detailed in Algorithm 1.

IV. PROPOSED ANTI-JAMMING METHOD AGAINST MULTIPLE JAMMERS

In SC2, users are simultaneously attacked by multiple jammers with different jamming policies. In this context, users cannot differentiate between the jamming policies of each jammed channel, and, thus, the proposed anti-jamming method in SC1 cannot be employed. We propose to estimate the channels that will be jammed in the next time slot using the jammers’ occupied channels in previous time slots. Once more, the interaction between the users and the jammers has a sequential nature, and, therefore, we also use RNNs in SC2. We consider two scenarios depending on whether or not users interfere with each other. Since we assume that the status of a frequency channel is solely determined by sensing the signal amplitude of that channel and no signal processing is performed, the users cannot distinguish between a jammed and an interfered channel.

In SC2, we consider an RNN for each user. In this context, after each GRU hidden layer, we use a FL with output size equal to the number of channels, as depicted in Fig. 4. Next, the output of the fully connected layer is fed into a Sigmoid function to normalize the output elements between zero and one. Afterward, the values that are higher than 0.5 are set to one and the ones lower than 0.5 to zero. The overall structure of the proposed network is detailed in Table I.
Algorithm 1: Proposed Method Against a Single Jammer

Initialize parameters: \( T, \tau \leftarrow 1, L, \) loss \( \leftarrow 0, \) LR \( \leftarrow 0.003, \)
\( \text{GRU} \_\text{Net} = \text{nn.GRU} \) (input_size = 2L, hidden_size = 64, num_layers = 4; 
\( \text{Fed} \_\text{forward} = \text{nn.Linear} \) (hidden_size = 64, output_size = 13; 
\( S_c = \text{zeros}(1, 2L + 1), S_t = \text{zeros}(1, 2L + 1); \)
\( J_p = \) ["Random", "Sweeping", "Reactive", "Combat"]; 
Training (Offline process): 
foreach Jamming policy in \( J_p \) do 
while \( i \leq T \) do 
User’s channel, \( U_c \leftarrow \text{rand}(1, L); \)
\( J_p \leftarrow \) Jammer selects its channels according to \( J_p; \)
i \( \leftarrow i + 1; \)
\( S_t(1:L) = \text{dec2bin}(U_c); \)
\( S_t(1:L + 1:2L) = \text{dec2bin}(J_p); \)
\( S_t(2L + 1) = \) Jamming policy index; 
\( S_c \leftarrow \text{Concat} \) (\( S_c, S_t); \)
end while 
end foreach 
foreach batch_idx, data in enumerate (data_loader) do 
x \( \leftarrow \) Data_set (batch_idx, 1:L); 
h \( \leftarrow \) GRU_net (x); 
on_out \( \leftarrow \) Fed_forward (h); 
loss \( \leftarrow \) nn.binary.CrossEntropyLoss (Data_set (batch_idx, 2:L+1), out_put); 
loss.backward() 
end foreach 
Testing (Online process): 
if \( i \leq 20 \) then 
User’s channel, \( U_c \leftarrow \text{rand}(1, L); \)
\( J_p \leftarrow \) Jammer selects its channels according to \( J_p; \)
i \( \leftarrow i + 1; \)
\( S_t(1:L) = \text{dec2bin}(U_c); \)
\( S_t(1:L + 1:2L) = \text{dec2bin}(J_p); \)
\( S_t(2L + 1) = \) Jamming policy index; 
\( S_c \leftarrow \text{Concat} \) (\( S_c, S_t); \)
else 
h \( \leftarrow \) GRU_net (S_c) (end \( \rightarrow 20; \) end, ); 
on_out \( \leftarrow \) Fed_forward (h); 
y_pred \( \leftarrow \) SoftMax (out_put); 
output_class \( \leftarrow \) argmax (y_pred); 
User’s channel \( \leftarrow \) the best channel out of free channels; 
\( J_p \leftarrow \) Jammer selects its channels according to \( J_p; \)
\( S_t(1:L) = \text{dec2bin}(U_c); \)
\( S_t(1:L + 1:2L) = \text{dec2bin}(J_p); \)
\( S_t(2L + 1) = \) Jamming policy index; 
\( S_c \leftarrow \text{Concat} \) (\( S_c, S_t); \)
end if 
i \( \leftarrow i + 1; \)
end while 

Algorithm 2: Proposed Method Against Multiple Jammers

Initialize parameters: \( T, \tau \leftarrow 1, L, \) loss \( \leftarrow 0, \) LR \( \leftarrow 0.003, \)
\( \text{GRU} \_\text{Net} = \text{nn.GRU} \) (input_size = 2L, hidden_size = 64, num_layers = 4; 
\( \text{Fed} \_\text{forward} = \text{nn.Linear} \) (hidden_size = 64, output_size = 1, L), 
\( S_c = \text{zeros}(1, 2L); \)
while Running do 
if \( i \leq T \) then 
User’s channel, \( U_c \leftarrow \text{rand}(1, L); \)
\( J_p \leftarrow \) Jammer selects its channels according to \( J_p; \)
\( S_t(1:L) = \text{dec2bin}(U_c); \)
\( S_t(1:L + 1:2L) = \text{dec2bin}(J_p); \)
\( S_c \leftarrow \text{Concat} \) (\( S_c, S_t); \)
else if \( i \geq T \) then 
Training (Online process): 
x \( \leftarrow \) Data_set (batch_idx); 
target \( \leftarrow \) Data_set (batch_idx + 1); 
h \( \leftarrow \) GRU_net (x); 
on_out \( \leftarrow \) Fed_forward (h); 
loss \( \leftarrow \) nn.binary.CrossEntropyLoss (target, out_put); 
loss.backward() 
end if 
end if 

Fig. 4. Proposed RNN for multiple jammers scenarios.

Similarly, the proposed anti-jamming method is divided into two phases, i.e., training and testing. In each time slot of the training phase, each user considers a vector with 2L elements, 
and saves its utilized channel in the first L elements while the occupied channels by jammers are saved in the second L elements (and other users in case of interference). Particularly, the \( i \)th element is set to one if the \( i \)th channel is occupied, otherwise it is set to zero. During training, we use a random channel selection policy for the users so as to generalize the results. However, the proposed training scheme can work with any other channel allocation methods. To provide the target data, the vector containing the occupied channels in the current time slot is used as the target of the previous time slots because we want to estimate the occupied channels in the next time slot.

In the testing phase, similar to the training phase, each user saves the occupied channels for the last \( b \) time slots. Then, the occupied channels of the last \( b \) time slots, including the current time slot are fed to the trained network, and the predicted occupied channels in the next time slot are generated by the network. The binary cross-entropy is utilized as the loss function to train the network. With the exception of a minor change where the outputs predicted occupied channels for the subsequent time slot, the testing procedure for SC2 is very similar to that for SC1. The proposed algorithm is detailed in Algorithm 2.

The proposed anti-jamming methods are based on GRU networks where the computational complexity order of LSTM or GRU networks is \( \mathcal{O}(F_o) [29] \), where \( F_o \) is 
\[
F_o = 4n_c^2 + 4n_t \times n_c + n_c \times n_o + 3 \times n_c
\]
(12)
n_c, n_t, and n_o are the number of memory cells, input units, and output units, respectively. Given that \( F_o \) is dominated by the \( n_c \times (n_c + n_o) \) factor, the computational complexity of the learning phase of LSTM or GRU networks can be obtained as \( \mathcal{O}(n_c \times (n_c + n_o)) \). In our considered networks, \( n_c = 2L \) and
n_o = C for the first considered scenario, and n_c = 2La and 

n_o = L for SC2, where a is the considered data batch size. 

Hence, the complexity order of the proposed RNN models for 

SC1 and SC2 is \( O(2La(2La+C)) \) and \( O(2La(2La+L)) \), respectively. For the tasks requiring a large number of output units and a large number of memory cells, learning LSTM models can be computationally expensive. However, in our considered cases, the numbers of inputs and outputs are values between 10 and 60, and a is 32. Moreover, the learning process of the proposed method in SC1 is offline and in the test process, users just need to multiply the inputs to the learning weights and sum the result with biases.

In the two anti-jamming techniques proposed in this work, the information about the jammed channels is required. To extract this information, each user senses all the frequency channels and flags channels where the amplitude of the baseband signal is higher than a specific threshold. In this context, the received jamming signal in a frequency channel is given as

\[
\chi_k = \gamma_k + \xi
\]

where \( \gamma_k \) is the amplitude of the jamming signal at the \( k \)th user’s side and \( \xi \) is the white Gaussian noise with power \( \delta^2 \).

Assuming that the detection threshold is \( \gamma \), a miss-detection happens when the noise amplitude is lower than \( \xi \leq (\gamma - \gamma_k) \) and a false alarm happens when \( \xi \geq \gamma \). Thus, the probability of miss-detection \( P_{md} \) and false alarm \( P_{fa} \) are given as

\[
P_{md} = Q\left(\frac{\gamma_k - \gamma}{\delta}\right)
\]

\[
P_{fa} = Q\left(\frac{\gamma}{\delta}\right)
\]

respectively. \( P_{fa} \) shows that the probability of false alarm only depends on the value of the threshold and noise power while \( P_{md} \) depends on the jamming signal power at the users’ side.

\( \gamma \) must be high enough to minimize \( P_{fa} \), and should not be too high since increasing \( \gamma \) increases \( P_{md} \). For instance, \( \gamma \) should satisfy \( (\gamma/\delta) = 2.32 \) to have \( P_{fa} = 0.01 \). Here, \( P_{md} = 0.01 \) for a JNR of \((||\gamma_k||^2/\sigma^2) = 13.35 \) dB. According to (15), the value of \( \gamma \) is adjusted according to the desired false alarm probability \( P_{fa} \), which only requires knowledge about the noise power \( \delta \). However, in order to find \( P_{fa} \) that guarantees an appropriate \( P_{md} \), information about the received jamming power to noise ratio at the users’ side is necessary. This information can be obtained by listening to the jammed channels in previous time slots. Specifically, users can find the noise power \( \delta \) by communicating with the AP, and, hence, they can calculate the average received jamming power by comparing the received powers in different channels with \( \delta \) since the sensed powers in the jammed channels are significantly higher than \( \delta \). Moreover, users can readjust the threshold when the performance of the proposed method decreases due to an inaccurate threshold setting. In addition, in the case of miss-detection, the jamming pattern can still be predicted from the jammers’ behavior in previous time slots. For both scenarios considered, we set \( (\gamma/\delta) = 2.8117 \) and JNR = 15 dB.

| Jammer type      | 30% | 40% | 50% | 60% | 70% |
|------------------|-----|-----|-----|-----|-----|
| Sweeping jammer  | 2   | 3   | 4   | 4   | 5   |
| Reactive jammer  | 1   | 1   | 1   | 1   | 1   |
| Random jammer    | 1   | 2   | 2   | 3   | 3   |
| Combat jammer    | 2   | 2   | 3   | 4   | 5   |

V. SIMULATION RESULTS

We now assess the proposed anti-jamming methods in both considered scenarios using extensive simulations. To this end, we define an evaluation metric, named STR, which quantifies the ratio of the successfully delivered packets to all the transmitted packets. For SC1, the proposed anti-jamming method is assessed by evaluating the STR and detection accuracy policies as a function of the elapsed time for the different jamming policies. We consider that random, combat, and sweeping jammers can simultaneously jam 1–4 channels. Moreover, we compare the output of the proposed anti-jamming method to a deep Q-learning (DQL)-based approach. Unless stated otherwise, RNN is the legend denotes the proposed method while DQL denotes the considered DQL benchmark for comparison. In addition, we compare the ER obtained by the proposed anti-jamming techniques to the calculated ER in (6) and (8). In SC1, simulation results are obtained using 12 channels. For SC2, we assume that each user is surrounded by a random, a combat, a sweeping, and a reactive jammer. The jammers’ specifications are provided in Table II. The proposed anti-jamming method is evaluated by illustrating the STR of 1–4 users when 30% to 70% of the spectrum is under jamming attacks. Moreover, results are compared to the case where DQL is employed for the users’ channel allocation. We present the STR obtained by the proposed anti-jamming method as a function of different numbers of channels and users. Furthermore, we show the performance of each jammer in terms of successful jamming attacks to clarify the impact of each jamming type separately. We consider a circular area with a radius of \( 20r_0 \) where users are randomly distributed. Then, the locations of the considered jammers around each user are randomly selected in a circular area with radius \( 2r_0 \) from it. In SC2, simulation results are obtained by considering 20 channels. Moreover, we set \( a = 20 \) and \( b = 30 \), \( T = 2000 \) and channels are assumed to follow Rayleigh fading, i.e., \( m = 1 \). The proposed models are implemented using the PyTorch Library. Four layers of GRU cells with a 64-D hidden state are used. Moreover, models are trained for 500 epochs with a minibatch size of 32 using the Adam optimizer with a learning rate of 0.003.

A. Anti-Jamming in SC1

In Fig. 5, we show the STR resulting from the proposed anti-jamming method for all the considered types of jammers. From this figure, we observe that after a few time slots, except for the random case, the obtained STR surpasses 90% and STRs of almost 100% are obtained after 20 time slots. The STR increases with time since the users are provided.
with more information from the jammers, which leads to jamming policy detection with a higher accuracy. As a result of a precise jamming policy detection, an appropriate countermeasure against the detected jammer is taken, resulting in a high STR. Lower STR is obtained in the context of random jammers since future channels cannot be estimated even for perfect jamming policy detection. As a result, users have to select their future channels randomly. In this context, increasing the number of jammed channels by the random jammer increases the probability of the user getting jammed. In addition, Fig. 5 shows that the proposed anti-jamming method outperforms the DQL-based method for all the considered cases, except for the Random jammer. Specifically, the proposed method converges within a few time slots while DQL needs more time slots to train the users.

Fig. 6 presents the detection accuracy for all the considered jamming policies. From this figure, we observe that, after just five time slots, the sweeping and combat jamming policies are detected perfectly. Detecting the policy of the random and reactive jammers requires more time slots due to the similarity between these jammers’ behavior in initial time slots.

In Fig. 7, we verify the ERs derived are (6) and (8) using simulations. Here, we consider that the number of jammed channels to varies between 1 and 4. The obtained ERs by the proposed anti-jamming method for all the considered jamming scenarios are shown. The presented graphs in order are: 1) ERs obtained by (6) with legend of TM; 2) simulation to verify (6) with legend of SM; 3) proposed anti-jamming method against sweeping and combat jammers with legend of PM-SC; 4) ERs obtained by (8) with legend of TR; 5) simulation to verify (8) with legend of SR; 6) proposed anti-jamming method against random jammer by assigning the channel with the highest channel gain to the user with legend of PM-R-M; and 7) proposed anti-jamming method against random jammer with random channel selection with legend of PM-R-R. Moreover, in Fig. 7, we present the ER achieved using the proposed anti-jamming method when the number of jammed channels is one by the legend of PM-RE.

Fig. 7. ER for SC1.

The results in Fig. 7 demonstrate that the calculated ERs perfectly match the simulation results, which validates the accuracy of our derivations. From this figure, we can also see that the ERs in the presence of reactive, sweeping, and combat jammers are close to the maximum achievable ERs. This, in turn, shows the effectiveness of the proposed anti-jamming method in recognizing the jamming policies and assigning high-quality channels to the user. Against the random jammer, the ER is lower than the other cases since the random jammer’s occupied channels in the next time slot are unpredictable. Moreover, for all the considered numbers of jammed channels by the random jammer, assigning the channel with the highest channel gain to the user provides a higher ER than the case in which the channel assignment is random.

B. Anti-Jamming in SC2

Fig. 8 presents the users’ STR as a function of the elapsed time slots. From Fig. 8, we can see that the users achieve an STR higher than 90% when 30% of the spectrum is
under jamming attacks. This value decreases to 80% when jammers jam 70% of the spectrum. The reason behind this performance degradation is that the users have less free channels to utilize. Moreover, for all the considered scenarios, the proposed method outperforms the DQL-based method with a significant gap.

In Fig. 9, we present the STR obtained by the proposed anti-jamming method for the multiple jammers scenario as a function of the number of channels when the reactive, random, combat, and sweeping jammers jam 1, 2, 3, and 4 channels per time-slot, respectively. Given that the proposed method is a fully distributed anti-jamming method, the results are independent of the number of users in the scenario when there is no interference between users. As a result, we present the STR of the single user for this case and 2–8 users for the case where users interfere with each other. These results show that increasing the number of channels increases the number of choices for each user, and consequently decreases the jamming rate. For the same reason, increasing the number of users decreases the STR of the users.

The STR of the users in case of interference is shown in Fig. 10. The performance of the proposed anti-jamming method decreases compared to the interference-free case. This is because the RNN is trained assuming that users choose their channels at random. As a result, each user randomly estimates the future behavior of other users during the testing process. Moreover, when users interfere with each other, fewer available channels remain for channel selection. For instance, in the four users scenario with 70% of the channels jammed, four users must select their channels out of six available channels while for the case in which the users do not interfere, each user has six available channels.

In Fig. 11, we present each jammer’s success rate individually. These results show that the reactive and sweeping jammers have constant jamming success rates for all the considered jamming ratios. Meanwhile, the random jammer’s success rate increases by increasing the jamming ratio. This
is because the reactive jammer has a fixed number of jammed channels for all the considered jamming ratios. Moreover, the future behavior of a sweeping jammer is predictable and, as a result, increasing the number of jammed channels in this type of jammer does not impact the success rate. However, it is impossible to predict the channels that the random jammer will choose in the following time slot. Therefore, increasing the number of jammed channels increases the random jammer’s success rate. Similar to random jammers, increasing the number of jammed channels by the combat jammer increases the success rate of the jammer due to randomness in the channel selection of the combat jammer. However, since the combat jammer jams the selected channels for a number of consecutive time slots, the increase in the success rate is lower than that of the random jammers.

Fig. 12 compares the ERs obtained by the proposed method, (6), and (7), with and without interference. Drawn graphs in order are related ERs obtained by: 1) Equation (6), denoted as TM; 2) Equation (7) for two users, denoted as TIM-2; 3) Equation (7) for three users, referred to as TIM-3; 4) Equation (7) for four users, denoted as TIM-4; 5) proposed method for the case where users do not interfere each other denoted as PM; 6) proposed method for two users scenario while users interfere each other specified with legend PMI-2; 7) proposed method for three users scenario while users interfere each other, denoted as PMI-3; and 8) proposed method for four users scenarios considering users interfere each other, denoted as PMI-4. When users do not interfere with each other, results show that the proposed method achieves ERs higher than 80% of the maximum achievable ER for all the considered numbers of jammed channels. Moreover, when 30% of the spectrum is being jammed, the ER is close to 90% of the maximum achievable ER. In the case where users interfere with each other, the discrepancy between the maximum achievable ER and the obtained ER by the proposed method is higher than the no interference case since fewer free channels are available for the channel selection of each user.

When the number of the jammers is not known, the proposed anti-jamming technique for multiple jammers scenarios should be employed since it can be applied to the single jammer scenario too. However, it is not optimal for the single jammer scenario, where it is best to employ the first proposed method. This challenge can also be addressed by using the proposed anti-jamming technique for the single user as the channel allocation policy of each user in the training process of the multiple jammers method. Given that the training process of the proposed anti-jamming method for the single jammer scenario is offline, each user can be provided with the trained network before interacting with the environment. In our implementation of the training phase for the multiple jammers scenario, we have assumed that users select their channels randomly while their channel selection can be changed to the proposed anti-jamming for a single jammer. In other words, when a jamming attack occurs, each user applies the proposed single user anti-jamming technique while also being trained for multiple jammers. Then, after the multiple jammers training process is terminated, users can compare the performance of both methods and adopt the one that delivers better performance.

In each simulation run, the locations of the nodes are generated and remain fixed until the end of the run. Despite considering fixed positions for nodes, our proposed methods are applicable and deliver the same results for mobile nodes for the following reasons. The mobility of the users changes the
JNRs at the users and, as a result, affects the accuracy of the occupied channels detection. Thus, in Fig. 13(a), we present the STR obtained by the proposed anti-jamming method for SC2 as a function of the JNR for different \( P_{fa} \). Results are obtained for a single user, a jamming ratio of 50%, and 20 available channels. Moreover, we show the variation of \( P_{md} \) as a function of the JNR in Fig 13(b). Results show that increase of the JNR up to 15 dB increases the user’s STR. However, increasing JNR after 15 dB does not impact on the STR. The reason behind this can be well understood by looking at the Fig 13(b). \( P_{md} \) decreases with the increase in the JNR up to 15 dB where at that point, the \( P_{md} \) reaches a value close to zero, allowing for extremely precise detection. The STRs at \( P_{fa} \) (decreasing the threshold level) up to 0.15; however, increasing \( P_{fa} \) after 0.15 decreases the STR. The reason behind the former is that when the JNR is low, the miss-detection rate decreases when the detection threshold increases and, as a result, a higher STR is achieved. However, when the JNR is high, the detection accuracy is sufficient for perfect detection, and raising \( P_{fa} \) raises the number of false alarms, which lowers the performance. The latter is justified by the fact that when \( P_{fa} \) is high, numerous false alarms occur, causing the trained network to make mistaken predictions and leading to a low STR.

Moreover, the following scenarios can be considered when nodes are mobile: a pair of jammer and user move at a fixed distance from each other, approach each other and move away from each other. In SC1, the average JNR is fixed since the distance between the user and the jammer is fixed. As a result, the situation is similar to the one we have considered. In the case of nodes approaching each other, the JNR increases, which results in higher detection accuracy. On the contrary, when a jammer moves away from a user, the JNR decreases on the user’s side. In this scenario, the detection accuracy decreases. However, the accuracy can be enhanced by increasing the number of samples. For instance, a single sample of the jamming signal in a given channel is

\[
χ_k = Υ_k + ξ.
\]

(16)

If the \( k \)th user samples two times from the jamming signal, the amplitudes of the received signals at the \( k \)th user are

\[
χ_{k1} = Υ_k + ξ_1 \quad χ_{k2} = Υ_k + ξ_2.
\]

(17)

The JNR value that the \( k \)th user obtains from the single sampling, i.e., \( χ_k = Υ_k + ξ \), is \( \left( \|Υ_k\|^2/σ^2 \right) \), while the JNR value that the \( k \)th user gets from \( χ_{k1} + χ_{k2} \) is \( \left( 2\|Υ_k\|^2/σ^2 \right) \), which means that increasing the number of samples to two increase the JNR by 3 dB. Thus, the JNR can be increased to 10 dB just by summing ten samples. Moreover, the JNRs are obtained in relation to jammers’ signals, which have a powerful source and consistently make an effort to approach the users. In addition, when the average JNR is very low such that it cannot be detected by the users, utilizing the jammed channels does not degrade the SINR of the users.

VI. CONCLUSION

In this article, we proposed RNN-based anti-jamming techniques against single jammer and multiple jammers. Two distinct scenarios based on the number of jammers in the network have been considered. In SC1, the network includes a user and a jammer capable of attacking with various jamming policies, while in SC2, we have assumed that multiple jammers attack multiple users with different jamming policies. Moreover, we studied two different cases based on whether users interfere with each other or not. For both of the considered cases, we calculated the maximum achievable ER. To evaluate the proposed anti-jamming methods, we have performed extensive simulations assuming four jamming policies. Moreover, we compared the obtained results with the case where DQL is employed. The results showed that against a single jammer, all the considered jamming policies are detected with high accuracy within a short period, and as a result of an accurate jamming-type detection, high STRs and ERs are obtained. Against multiple jammers, STRs and ERs near 80% are obtained when 70% of the spectrum is under jamming attacks. These values rise to 90% when 30% of the spectrum is under jamming attacks. Moreover, for all the considered numbers of users and jamming ratios, the proposed anti-jamming technique outperforms the DQL algorithm with significant gaps.


APPENDIX B
PROOF OF PROPOSITION 2

The proof of Proposition 2 is similar to that of Proposition 1 with the only difference that the channel $h_{ek}$ is chosen randomly. Thus, the PDF and CDF of $x = |h_{ek}|^2$ follow (20) and (21), respectively. Substituting the CDF of $x$ into (24) leads to

$$R_{ek} = \frac{\Omega_k}{\delta^2 \Gamma(m) \ln 2} \int_0^\infty \Gamma(m, \frac{\delta x}{\Omega_k}) \frac{\Omega_k}{\delta x} dx$$  \hspace{1cm} (24)$$

and since $\langle 1 + \frac{1}{\Omega_k} \frac{\delta x}{\Omega_k} \rangle = G_{1,1}^{1,1}(\frac{\Omega_k}{\delta x} \frac{\delta x}{\Omega_k}) \bigg|_0^0$ [30, eq. (8.4.2.5)] and $\Gamma(m, \{m/\lambda\}) = G_{2,0}^{2,0}(m/\lambda, x) \bigg|_0^1$ [30, eq. (8.4.16.2)], (24) can be rewritten as

$$R_{ek} = \frac{\Omega_k}{\delta^2 \Gamma(m) \ln 2} \int_0^\infty G_{1,1}^{1,1} \bigg|_0^0 \frac{\Omega_k}{\delta x} \bigg|_0^1 dx$$

(25)

By using the relationship in [31, eq. (7.811)], (8) can be obtained, which completes the proof of Proposition 2.

REFERENCES

[1] A. Pourranjbar, G. Kaddoum, and W. Saad, “Jamming pattern recognition over multi-channel networks: A deep learning approach,” 2021, arXiv:2112.11222.

[2] Y. Xuan, Y. Shen, N. P. Nguyen, and M. T. Thai, “A trigger identification service for defending reactive jammers in WSN,” IEEE Trans. Mobile Comput., vol. 11, no. 5, pp. 793–806, May 2012.

[3] S. Nan, S. Braham, C. A. Kamhoua, and N. O. Leslie, “Mitigation of jamming attacks via deception,” in Proc. IEEE Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC), London, U.K., Aug. 2020, pp. 1–6.

[4] A. Pourranjbar, G. Kaddoum, A. Ferdowsi, and W. Saad, “Reinforcement learning for defending reactive jammers in wireless networks,” IEEE Trans. Commun., vol. 69, no. 6, pp. 3682–3697, Jun. 2021.

[5] S. D’Oro, L. Galluccio, G. Morabito, S. Palazzo, L. Chen, and F. Martignon, “Defeating jamming with the power of silence: A game-theoretic analysis,” IEEE Trans. Wireless Commun., vol. 14, no. 5, pp. 2337–2352, May 2015.

[6] A. Pourranjbar, G. Kaddoum, and K. Aghababaiyan, “Deceiving-based anti-jamming against single-tone and multitone reactive jammers,” IEEE Trans. Commun., vol. 70, no. 9, pp. 6133–6148, Sep. 2022.

[7] F. Slimeni, B. Scheers, Z. Chimouro, V. L. Nir, and R. Atia, “A modified Q-learning algorithm to solve cognitive radio jamming attack,” Int. J. Embedded Syst., vol. 10, no. 1, pp. 41–51, 2018.

[8] F. Yao and L. Jia, “A collaborative multi-agent reinforcement learning anti-jamming algorithm in wireless networks,” IEEE Wireless Commun. Lett., vol. 8, no. 4, pp. 1024–1027, Aug. 2019.

[9] I. Elleuch, A. Pourranjbar, and G. Kaddoum, “A novel distributed multi-agent reinforcement learning algorithm against jamming attacks,” IEEE Commun. Lett., vol. 25, no. 10, pp. 3204–3208, Oct. 2021.

[10] Q. Qu, S. Wei, S. Liu, J. Liang, and J. Shi, “JRNNet: Jamming recognition networks for radar compound suppression jamming signals,” IEEE Trans. Veh. Technol., vol. 69, no. 12, pp. 15035–15045, Dec. 2020.

[11] G. Shao, Y. Chen, and Y. Wei, “Convolutional neural network-based radar jamming signal classification with sufficient and limited samples,” IEEE Access, vol. 8, pp. 80588–80598, 2020.

[12] Z. Wu, Y. Zhao, Z. Yin, and H. Luo, “Jamming signals classification using convolutional neural network,” in Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT), Bilbao, Spain, Jun. 2017, pp. 062–067.

[13] Y. Wang, B. Sun, and N. Wang, “Recognition of radar active-jamming through convolutional neural networks,” J. Eng., vol. 2019, no. 21, pp. 7695–7697, 2019.

[14] Y. Cai, K. Shi, F. Song, Y. Xu, X. Wang, and H. Luan, “Jamming pattern recognition using spectrum waterfall: A deep learning method,” in Proc. IEEE 5th Int. Conf. Comput. Commun. (ICCC), Chengdu, China, Dec. 2019, pp. 2113–2117.

[15] A. Garnaev, A. Petropulu, W. Trappe, and H. V. Poor, “A multi-jammer game with latency as the user’s communication utility,” IEEE Commun. Lett., vol. 24, no. 9, pp. 1899–1903, Sep. 2020.

[16] A. Garnaev, A. Petropulu, W. Trappe, and H. V. Poor, “A multi-jammer power control game,” IEEE Commun. Lett., vol. 25, no. 9, pp. 3031–3035, Sep. 2021.

[17] N. Van Huynh, D. T. Hoang, D. N. Nguyen, E. Dutkiewicz, and M. Mueck, “Defeating smart and reactive jammers with unlimited power,” in Proc. IEEE Wireless Commun. Netw. Conf., Seoul, South Korea, May 2020, pp. 1–6.

[18] I. Loth, D. Niyato, S. Sun, H. T. Dinh, Y. Li, and D. L. Kim, “Protecting multi-function wireless systems from jammers with backscatter assistance: An intelligent strategy,” IEEE Trans. Veh. Technol., vol. 70, no. 11, pp. 11812–11826, Nov. 2021.

[19] V. Le Nir and B. Scheers, “Multiple jammer localization and transmission power estimation for radio environment map,” in Proc. Int. Conf. Mil. Commun. Inf. Syst. (ICMCIS), Warsaw, Poland, May 2018, pp. 1–5.

[20] S. Bhamidipati and G. X. Gao, “Simultaneous localization of multiple jammers and receivers using probability hypothesis density,” in Proc. IEEE/ION Position Navigation Symp. (PLANS), Monterey, CA, USA, Apr. 2018, pp. 940–944.

[21] M. Juhlin and A. Jakobsson, “Localization of multiple jammers in wireless sensor networks,” in Proc. 29th Eur. Signal Process. Conf. (EUSIPCO), Dublin, Ireland, Aug. 2021, pp. 1596–1600.

[22] Z. Li, Y. Lu, X. Li, Z. Wang, W. Qiao, and Y. Liu, “UAV networks against multiple maneuvering smart jamming with knowledge-based reinforcement learning,” IEEE Internet Things J., vol. 8, no. 15, pp. 12289–12310, Aug. 2021.

[23] Y. Wu, W. Yang, X. Guan, and Q. Wu, “Energy-efficient trajectory design for UAV-enabled communication under malicious jamming,” IEEE Wireless Commun. Lett., vol. 10, no. 2, pp. 206–210, Feb. 2021.

[24] Y. Wu, X. Guan, W. Yang, and Q. Wu, “UAV swarm communication under malicious jamming: Joint trajectory and clustering design,” IEEE Wireless Commun. Lett., vol. 10, no. 10, pp. 2264–2268, Oct. 2021.

[25] B. Gingras, A. Pourranjbar, and G. Kaddoum, “Collaborative spectrum sensing in tactical wireless networks,” in Proc. IEEE ICC, Dublin, Ireland, Jun. 2020, pp. 1–6.

[26] I. Lundén, M. Motani, and H. V. Poor, “Distributed algorithms for sharing spectrum sensing information in cognitive radio networks,” IEEE Trans. Wireless Commun., vol. 14, no. 8, pp. 4667–4678, Aug. 2015.

[27] K. Cho et al., “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” 2014, arXiv:1406.1078.

[28] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” 2014, arXiv:1412.5555.

[29] H. Sak, A. W. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in Proc. Interspeech, 2014, pp. 1–5.

[30] A. Prudnikov, Y. Brychkov, and O. Marichev, Integrals and Series, Volume 3: More Special Functions. Boca Raton, FL, USA: CRC Press, 1999.

[31] I. S. Gradshteyn and I. M. Ryzhik, Tables of Integrals, Series, and Products, 7th ed. New York, NY, USA: Academic, 2007.
Georges Kaddoum (Senior Member, IEEE) received the bachelor’s degree in electrical engineering from the École de technologie supérieure de techniques avancées, Brest, France, in 2005, the M.S. degree in telecommunications and signal processing (circuits, systems, and signal processing) from the Université de Bretagne Occidentale and Telecom Bretagne, Brest, in 2005, and the Ph.D. degree (Hons.) in signal processing and telecommunications from the National Institute of Applied Sciences, University of Toulouse, Toulouse, France, in 2009.

He is currently a Professor and a Tier 2 Canada Research Chair with the École de technologie supérieure (ÉTS), Université du Québec, Montréal, QC, Canada. He was awarded the ÉTS Research Chair in physical-layer security for wireless networks in 2014. Since 2010, he has been a Scientific Consultant in the field of space and wireless telecommunications for several U.S. and Canadian companies. He has published over 300 journals, conference papers, two chapters in books, and has eight pending patents. His recent research activities cover wireless communication networks, tactical communications, resource allocations, and security.

Prof. Kaddoum received the Best Papers Awards at the 2014 IEEE International Conference on Wireless and Mobile Computing, Networking, Communications, with three coauthors, and at the 2017 IEEE International Symposium on Personal Indoor and Mobile Radio Communications, with four coauthors, the IEEE TRANSACTIONS ON COMMUNICATIONS Exemplary Reviewer Award for the years 2015, 2017, and 2019, the Research Excellence Award of the Université du Québec in the year 2018, the Research Excellence Award from TS in 2019, in recognition of his outstanding research outcomes, and the IEEE Technical Committee on Scalable Computing Award for excellence in scalable computing for the year 2022. He is currently serving as an Area Editor for the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON COMMUNICATIONS, and IEEE COMMUNICATIONS LETTERS.

Walid Saad (Fellow, IEEE) received the Ph.D. degree from the University of Oslo, Oslo, Norway, in 2010.

He is currently a Professor with the Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA, where he leads the Network Science, Wireless, and Security Laboratory. He has authored/coauthored of eleven conference best paper awards at WiOpt in 2009, ICIMP in 2010, IEEE WCNC in 2012, IEEE PIMRC in 2015, IEEE SmartGridComm in 2015, EuCNC in 2017, IEEE GLOBECOM in 2018, IFIP NTMS in 2019, IEEE ICC in 2020 and 2022, and IEEE GLOBECOM in 2020. He has coauthored the 2019 IEEE Communications Society Young Author Best Paper and of the 2021 IEEE Communications Society Young Author Best Paper. His research interests include wireless networks (5G/6G/beyond), machine learning, game theory, security, unmanned aerial vehicles, semantic communications, cyber–physical systems, and network science.

Dr. Saad is the recipient of the NSF CAREER Award in 2013, the AFOSR Summer Faculty Fellowship in 2014, the Young Investigator Award from the Office of Naval Research in 2015, the 2015 and 2022 Fred W. Ellersick Prize from the IEEE Communications Society, the 2017 IEEE ComSoc Best Young Professional in Academia Award, the 2018 IEEE ComSoc Radio Communications Committee Early Achievement Award, the 2019 IEEE ComSoc Communication Theory Technical Committee, and the Dean’s Award for Research Excellence from Virginia Tech in 2019. He was named the Stephen O. Lane Junior Faculty Fellow at Virginia Tech from 2015 to 2017, and was named the College of Engineering Faculty Fellow in 2017. He was also an IEEE Distinguished Lecturer from 2019 to 2020. He currently serves as an Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING and the IEEE TRANSACTIONS ON COGNITIVE COMMUNICATIONS AND NETWORKING, an Associate Editor-in-Chief for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (JSAC) Special Issue on Machine Learning for Communication Networks, and an Editor-at-Large for the IEEE TRANSACTIONS ON COMMUNICATIONS. He is the Editor-in-Chief for the IEEE TRANSACTIONS ON MACHINE LEARNING IN COMMUNICATIONS AND NETWORKING.