Machine Learning Based Protocol Classification in Unlicensed 5 GHz Bands

Wenhan Zhang and Marwan Krunz

Dept. Electrical & Computer Engineering, University of Arizona, Tucson, AZ
{wenhanzhang, krunz}@email.arizona.edu

***Abstract*—To monitor RF activity and efficiently coordinate channel access for heterogeneous wireless systems over a shared channel, it is important to be able to classify observed transmissions accurately without decoding them. In this paper, we propose novel recurrent neural network (RNN) architectures for signal classification, considering as a use case on interleaving-based spectrum sharing model for Wi-Fi, LTE-LAA, and 5G-NRU over the unlicensed 5 GHz bands. Several classifiers are presented, which take raw in-phase/quadrature (I/Q) samples as input. First, we examine Simple RNNs, Long Short-term Memory (LSTM) networks, and Gated Recurrent Units (GRU) networks for protocol classification. These RNNs are used to capture the unique features in observed signals. To further improve the classification accuracy, we extend the RNN designs into a bidirectional structure, allowing an RNN cell to learn the temporal dependence in the waveform in both forward and backward directions. Bidirectionality can effectively increase the amount of information and the context available to the neural network. We then extend our designs to multi-layer RNNs, which allow the classifier to capture temporal correlations at multiple time scales, hence increasing the network's computational capacity. Finally, we propose further enhancements to reduce the over-fitting problem in RNN training, including regularization, recurrent weight constraints, and rate halving. Our simulation results show that the multi-layer and bidirectional designs can effectively improve the accuracy of the RNN-based RF signal classifier. Combining the two features, an RNN structure can achieve more than 92% accuracy in our protocol classification problem.**

***Index Terms*—Deep learning, signal classification, coexistence, recurrent neural networks, dynamic spectrum access**

I. Introduction

The demand for wireless capacity continues to outgrow spectrum availability, especially at low and mid bands (e.g., sub-6 GHz). To efficiently utilize the congested spectrum, various spectrum-sharing architectures have been proposed [1]. For example, in the Citizens Broadband Radio Service (CBRS), a three-tiered spectrum authorization access system is employed, which enables commercial users to share spectrum with incumbent federal and non-federal users [2]. A dynamic frequency selection approach was adopted for the Unlicensed National Information Infrastructure (UNII) bands, permitting LTE license assisted access (LAA) and 5G unlicensed (NR-U) cellular technologies to share the unlicensed spectrum with Wi-Fi devices [3], [4] in sub-6 GHz bands. The coexistence of various waveforms inevitably introduces interference among users. Interference may originate from legitimate devices or may come from adversarial systems that aim at jamming or capturing the channel and preventing legitimate devices from accessing it. Therefore, it is critical for network coordinators to be able to classify observed signals for the purpose of ensuring fair access and detecting nonconforming and adversarial behavior.

Common spectrum sensing approaches are usually based on energy detection, maximum likelihood estimation, and cyclostationarity [5]. Many of these approaches require the receiver to know the protocol semantics as well as the transmitter parameters (e.g., frequency offset). Such methods are often designed to detect the signal of a particular protocol. However, in some spectrum sharing scenarios, multiple heterogeneous protocols may contend for the shared spectrum. For example, in the unlicensed 5 GHz bands, Wi-Fi, LTE LAA, and 5G NR-U devices can share the common spectrum [6], [7]. Unless a given contending device is equipped with multiple radios, it is difficult to identify the waveforms of different signal protocols. In addition, the conventional methods require listening for a certain period when receiving the waveforms to estimate the periodicity and calculate the correlation with the known sequences (e.g., the preamble). Nonetheless, such listening time can be insufficient for the fast adaption between different protocols, especially in the dynamically shared bands. To address the above limitations, we propose a deep neural network (DNN) based framework for accurate and fast signal sensing and classification in multi-protocol coexistence scenarios.

If the modulation and coding scheme (MCS) of the captured signal is unknown, then a protocol classifier would have to rely on down-converted baseband signals to be used as input. In recent years, different neural networks have been designed to classify RF signals based on baseband in-phase/quadrature (I/Q) samples. In [8], [9], the authors used the moving kernel of a convolutional neural network (CNN) to capture features hidden in a segmented sequence of I/Q samples. However, CNNs are not particularly effective at capturing temporal dependencies. In contrast, a recurrent neural network (RNN) can capture the memory (time dependency) in the data, which explains its widespread use in forecasting problems, such as language modeling, speech recognition, and trajectory prediction of moving objects. The authors in [10] applied a Long Short-term Memory (LSTM) network, a type of RNNs, for *automatic modulation classification*. Their proposed design outperforms a CNN classifier at high SNRs. In [11], the authors used both CNN and LSTM networks to generate

adversarial RF waveforms that successfully mislead legitimate classifiers. Most existing DNN-based RF classification efforts focus on modulation scheme identification. However, the three examined protocols have several modulation schemes in common, and they may operate over the same bandwidth. Hence, modulation classification techniques cannot differentiate between the waveforms of such protocols, which calls for new machine learning-based classifiers that capture other protocol-related embedded features in the observed transmissions.

A traditional protocol detection approach relies on the time-correlated view of the received sequences, which is similar to an RNN that can exhibit temporal dynamic behavior over the input sequence. Therefore, we investigate the application of RNNs for heterogeneous protocol classification over a shared spectrum, focusing on Wi-Fi, LTE-LAA, and 5G NR-U in the unlicensed 5 GHz bands as an example. We consider an *interleaving spectrum sharing approach*, where any but only one of the three coexisting technologies can be active at a time. Starting first with a basic two-layer RNN model, we show that this model can achieve around 71% classification accuracy, on average. To improve the classification accuracy, we then consider more advanced RNN models, involving bidirectional and multi-layer (hierarchical) gated structures. Unlike basic RNN models, these advanced networks have gates that balance the impact of the most recent input and the trained state during the recurrence. In addition, our bidirectional RNN structures allow information from the past (backward) and future (forward) states to be simultaneously used during the training. Such bidirectional design helps the RNN detect the backward dependency that the forward structure cannot capture. Our results show that the bidirectional design can increase not only the accuracy but also the precision and recall. In our hierarchical RNN designs, the classification outcomes of a lower RNN layer are used as input to train the next upper layer, which further increases the computational capability of the RNN-based classifier.

On the other hand, the multi-layer and bidirectional structures increase the neural network's complexity and give rise to possible model over-fitting. To solve these problems, we propose further enhancements, including regularization, dropout, recurrent weight constraints, and learning rate halving. Matlab-based simulations of the three coexisting protocols over the 5 GHz band were conducted for various proposed designs. They show that the average classification accuracy as a result of the novel RNN structures exceeds 92%.

II. Basic RNN Models

A. Simple RNN

An RNN is a type of neural networks that is designed for sequential processing. At each time step j, $j = 1, 2, ...$, a basic RNN cell takes sequence x_j as input and updates the corresponding learnable hypothesis parameter θ. In contrast to CNNs, RNNs can capture temporal dependencies over the entire input. A typical RNN model can be unfolded over j, where the current hidden state h_j is updated based on the previous state h_{j-1} and the external input x_j. Formally, $h_j = f(h_{j-1}, x_j; \theta)$, where f is a mapping function. For a finite number of time steps, state h_j can include the previous state input in a recursive manner:

$$h_j = f(h_{j-1}, x_j; \theta) \ = f(f(h_{j-2}, x_{j-1}), x_j; \theta) = \cdots \quad (1)$$

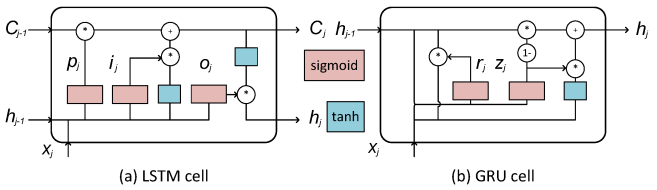


Fig. 1. Cell architecture for LSTM and GRU networks.

B. LSTM Network

An LSTM network builds on the simple RNN architecture but adjusts the updating procedure of the hidden state. We define the cell as the unit for state update, as shown in Figure 1(a). At each j, the cell output at the previous time step, h_{j-1}, is combined with x_j to form the input to the current cell. The state of the cell is C_j, which records the system memory and gets updated at each j. In contrast to simple RNNs, an LSTM network uses several gates, including an input gate (i_j), an output gate (o_j), and a forget gate (p_j). These gates are defined in Equation (2) below. They are used to control the effect of the input and output information through a sigmoid (σ) function:

$$\begin{aligned} i_j &= \sigma(W_i x_j + U_i h_{j-1} + b_i) \\ o_j &= \sigma(W_o x_j + U_o h_{j-1} + b_o) \\ p_j &= \sigma(W_p x_j + U_p h_{j-1} + b_p). \end{aligned} \quad (2)$$

In (2), W_i, W_o, and W_p are weights assigned to the three gates; U_i, U_o, and U_p are the corresponding recurrent weights; and b_i, b_o, and b_p are the bias values of the three gates. Similar to the gate function, x_i and h_{j-1} are combined to update the intermediate cell state $\widetilde{C}_j$. Instead of a sigmoid, the inputs are processed by a hyperbolic tangent function (tanh) that generates an output value between -1 and 1:

$$\widetilde{C}_j = \tanh(W_c x_j + U_c h_{j-1} + b_c). \quad (3)$$

After knowing C_{j-1} and $\widetilde{C}_j$, the cell state at time j is updated by the forget gate and the input gate as follows:

$$C_j = p_j * C_{j-1} + i_j * \widetilde{C}_j. \quad (4)$$

The output of the cell h_j, which will be used at time $j+1$, is obtained from by the element-wise product of the output gate and the hyperbolic tangent functioned C_j:

$$h_j = o_j * \tanh(C_j). \quad (5)$$

C. GRU Model

The main difference between the GRU model and the LSTM network is that GRU models use the same unit to control the input gate and the forget gate factor, as shown in Figure 1(b). Therefore, a GRU has only two gates: a reset gate r_j and an

update gate z_j. At each step j, these gates are calculated as follows:

$$\begin{aligned} r_j &= \sigma\left(W_r x_j + U_r h_{j-1} + b_r\right) \\ z_j &= \sigma(W_z x_j + U_z h_{j-1} + b_z). \end{aligned} \tag{6}$$

Similar to LSTM structure, W_r and W_z are gate weights; U_r and U_z are recurrent weights; and b_r and b_z are bias vectors. r_j determines the combination rate of the current input and the previous state's output, while z_j defines the amount of the previous state's output that will be used in the current step. As depicted in Figure 1(b), the intermediate hidden state $\widetilde{h}_j$ is determined by x_j and h_{j-1}, where h_{j-1} is processed by r_j before tanh. Thus, $\widetilde{h}_j$ can be expressed as:

$$\widetilde{h}_j = \tanh(W_h x_j + U_h(r_j * h_{j-1}) \ + \ b_h). \tag{7}$$

A GRU does not have the extra output gate to apply the nonlinearity as in (5). Hence the output of the cell is the direct gated combination of h_{j-1} and $\widetilde{h}_j$:

$$h_j = (1 - z_j) * h_{j-1} + z_j * \widetilde{h}_j. \tag{8}$$

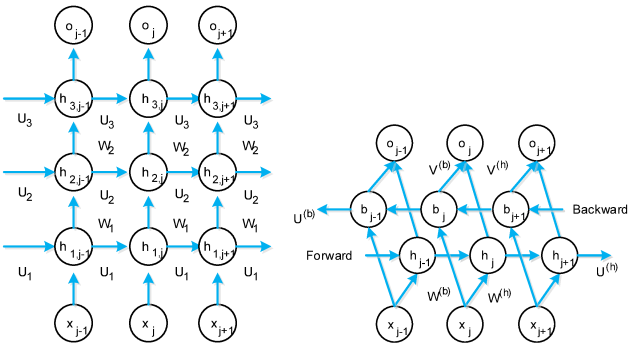
III. Multi-layer and Bidirectional RNNs

Although the basic RNN structures described in the previous section have the unique ability to capture correlations in the signal, capturing long-term dependencies is still challenging due to the gradient vanishing and exploding problems. In addition, long-term correlations tend to be subdued by small perturbations caused by short-term variations in the input. In this paper, we explore the use of multi-layer and bidirectional RNN-based to increase the computational capacity of the classifier. We also propose further enhancements to regularize and constrain the recurrent structure so as to balance the over-fitting problem raised by the higher model complexity. Combining the above three aspects, we apply the designs to our protocol classification problem.

A. Multi-layer RNN Structure

We consider a stacked RNN architecture in which the output of a RNN layer is used as input to the next-upper RNN layer, as shown in Fig. 2(a). The layer can be any RNN structure, such as a standard RNN (i.e., the SimpleRNN in *TensorFlow*), a LSTM [10], or a GRU. During training, the classification outcomes of a lower layer are used as inputs to train the next upper layer. Thus, the output at the final layer (i.e., classification layer) is expected to achieve higher classification accuracy than any lower-layer network. This stacked architecture captures temporal correlations at different time scales without using too many input samples. It allows the lower layer to transform the raw input into a more suitable format (e.g., remove the unrelated samples and disturbances).

An RNN cell can be regarded as a hypothesis that updates according to (1). The output $\bar{y}_j$ is generated by the $\phi(h_j)$, where ϕ is the activation function. After that, the loss function L can be calculated based on $\bar{y}_j$ and desired prediction y_j. To minimize this loss function, we train the model and update θ at each iteration by the back-propagated gradient: $g = \frac{1}{J}\nabla_\theta \sum_{j=1}^{J} L(\phi(f(h_{j-1}, x_j; \theta)), y_j)$. When it comes to the multi-layer RNN structure, the output of the hidden layer in layer l at time j can be computed from the hidden layer output of both lower layer at current time j $(h_{l-1,j})$ and the same layer at previous time $(h_{l,j-1})$. Combined with the hypothesis parameter θ, the hidden layer output can be formulated as:



(a) Multi-layer RNN (b) Bidirectional RNN

Fig. 2. Computational graph for recurrent network with multi-layer and bidirectional structures.

$$h_{j,l} = f(h_{j-1,l}, h_{j,l-1}; \theta). \tag{9}$$

The unfolding of RNN involves two-dimensional calculation in layer and time, which leads to a different information flow than a regular RNN structure. The gradient propagated through layer and time is used to calculate the weights of the input and previous hidden state. A typical gradient of the weight matrix from hidden unit g_θ can be computed by its Jacobian matrix $\frac{\partial h_{j,l}}{\partial \theta}$ and the partial derivative vector of the loss $g_{h_{j,l}}$: $g_\theta = \frac{\partial h_{j,l}}{\partial \theta} g_{h_{j,l}}$. As a result, the singular values of the Jacobian matrix decide the magnitude of gradients. A small value will attenuate the gradient and result in a vanishing problem. In contrast, a large value results in gradient explosion. For a deep RNN structure, the gradient of the hidden layer can be affected by the previous layer and time. Therefore, the hidden unit gradient can be expanded as:

$$g_{h_{j,l}} = \frac{\partial h_{j,l+1}}{\partial h_{j,l}} g_{h_{j,l+1}} + \frac{\partial h_{j+1,l}}{\partial h_{j,l}} g_{h_{j+1,l}}. \tag{10}$$

To reduce the impact of extreme gradient updates, we consider the adaptive moment optimizer, also known as ADAM. In this optimization algorithm, the DNN first estimates the first- and second-order moment to correct the bias. The step gradient is the same as in stochastic gradient descent (SGD), but two more moment variables are added: s and q. The estimated first moment is updated as $s = \rho_1 s + (1 - \rho_1)g$. Similarly, the estimated second moment can be updated as $q = \rho_2 q + (1 - \rho_2)g * g$, where $*$ operator is the element-wise product of a matrix. To correct the initialization bias, we introduce the correction terms of the first and second moments which are given by: $\hat{s} = \frac{s}{1-\rho_1}$, and $\hat{q} = \frac{q}{1-\rho_2}$, where ρ_1 and ρ_2 is the exponential decay rate and can be decided in the initial steps. We set $\rho_1 = 0.9$ and $\rho_2 = 0.999$ as the default values. Then, the parameter update can be decided by the moment estimation: $\Delta\theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{q}} + \delta}$, where $\delta = 0.00001$

is a small constant for stabilization, and ϵ is the step size (set to 0.001). The momentum is incorporated into the update for θ, and helps adapt the updating step according to the gradient. Besides, this adaption also makes ADAM not too sensitive to the initial learning rate, so it is more robust to the choice of hyperparameters than SGD.

B. Bidirectional RNN Structure

Bidirectional RNNs connect two hidden layers of opposite directions to the same output. With this form of generative deep learning, the output layer receives information from the past (backward) and future (forward) states simultaneously. This leads to improved accuracy because classification is now based on merging the results from both directions, i.e., chronological and inverse chronological orders. Such bidirectionality makes the network non-causal, whereby future information can influence the current decision. However, this non-causality is applied only during the training of the RNN network. Once the network has been trained, real-time classification (testing part) is performed using only currently received samples.

In our case, outputs are possible signal protocols. The loss function is obtained from the output of the hidden layer and the actual value of y. Therefore, the output o can be regarded as the unnormalized log probabilities of each possible value of the labels. The total loss for a given sequence x and the corresponding label y can be represented as the sum of losses over all time steps:

$$L = L\left(x_1, ..., x_J, y_1, ..., y_J\right) = \sum_{1}^{J} L_j = \sum_{1}^{J} \log\ p_{model}(y_j | x_1, ..., x_j) \quad (11)$$

where $p_{model}(y_j|x_1, ..., x_j)$ is the probability loss calculated from y_j and model's input $x_1, ..., x_j$. However, we still consider a causal structure, where at the time j, only the samples prior to j make contributions to f. Because the signal waveform contains correlations from the sequences after the current input x_j, we include another intermediate layer beginning from the end of the sequence, as drawn in Figure 2(b). In this figure, h_j is the state of the sub-recurrent layer that moves forward and b_j is the state of the sub-recurrent layer that moves backward. Accordingly, the output unit can benefit from both directions and compute the outcome based on both the past and the future, i.e.,

$$o_j = f(V_j^{(h)} h_j, V_j^{(b)} b_j) \quad (12)$$

where f function is the mapping function that combines the output sequences. In our proposed structure, the outputs for both the forward and backward layers are calculated in a recurrent way. Unlike (11), the backward weight matrix $V_j^{(b)}$ is updated by the input after time j, thus including the impact of the future inputs.

Note that $V_j^{(h)} h_j$ and $V_j^{(b)} b_j$ are only used to train f with parameter θ. When the training is finished, the predicted label can be expressed as: $\bar{y}_j = \phi(f(x_j; \theta))$. For the predicted label is only regarded to the current input x_j, the testing system is still causal. When applying a bidirectional RNN network model for protocol classification in shared-spectrum environments, the states of various cells need to be updated in both directions simultaneously to ensure that each unit receives updates from the whole sequence.

C. Further Enhancements

Our proposed multi-layer and bidirectional RNN structures help capture more features in input sequences but they also increase the complexity of the classifier. They may lead to a over-fitting situation during the training process. Additionally, RNNs in general may face a gradient exploding problem due to the reuse of sequential states. As shown in (2) and (6), the recurrent weight matrix U^j at step j is updated using nonlinear activation functions, which could result in weights and gradients exploding when j is large. We propose several further enhancements to constrain these side effects:

1) Regularization and Dropout: Regularization can be used to tune the weight matrix by adding additional penalty terms into the loss function. The added terms control excessively fluctuating function values and associate with the weight coefficients such that the coefficients do not take extreme values. Dropout refers to randomly deleting connections between computational units (neurons) during the training process of a DNN. It helps reduce the reliance on specific units and dilution of the weights. Using both approaches, RNN can reduce the generalization error between the evaluation set and the training set.

2) Recurrent Weight Constraints: Recurrent weight constraints check the norm of the recurrent weights and rescale them below a pre-defined threshold. Weight constraints are per-variable projection functions, applied to the target weight matrix after each gradient update during the training. Such constraints force each hidden state to have a norm that is less than or equal some desired value.

3) Learning Rate Halving: The learning rate is bounded by the step size parameter ϵ. This setting is suitable at the beginning of the training, but ϵ may be too large to detect small changes well into the training process. Learning rate halving can solve this problem, where we associate ϵ with the epoch number. During the initial training epochs, the weight matrix differs a lot from the desired one, allowing the back-propagated gradients to be more significant. When updating the weights, the weight matrix gets closer to the near-optimal one, and the corresponding gradients will be limited to a smaller value. This way, our RNN models can still update from the small gradients even after many iterations in the training process.

IV. Performance Evaluation

A. Data Generation

We use *Matlab Communication* and *5G Toolboxes* to generate waveforms of LTE, Wi-Fi, and 5G NR protocols. A set of signal features supported by Matlab, including channel

bandwidth, modulation schemes, I/Q imbalance, DC offset, and subcarrier spacing are varied. Through these features, we can generate diverse waveforms under different parameter settings. Of the various possible features, we consider the baseband I/Q samples at the receiver (with added noise) as input to the classifier. I/Q samples can be easily obtained before decoding the signal, and they provide a rich representation of the waveform. By applying a sliding window, these samples are divided into multiple sequences, each consisting of 512 I/Q pairs. These sequences are used as datasets to train and test various classifiers. Approximately 15,000 of such segments were obtained, split into 70% for training and 30% for testing. In this paper, we assume all protocols operate on the same center frequency and have a channel bandwidth of 20 MHz. This classification problem is more challenging because the frequency information or spectrogram can only provide a limited contribution to distinguish signal types. In addition, we assume an AWGN channel for all transmission. The Wi-Fi waveform is transmitted by generating baseband samples of 802.11ac (VHT) with BPSK modulation and $1/2$ code rate. LTE waveforms are generated assuming downlink reference measurement channel with R.9. This waveform uses 64 QAM modulation. We also generate 5G waveforms using 5G downlink fixed reference channel under QPSK modulation and a code rate of $1/3$, with a subcarrier spacing of 15 kHz.

B. Impact of Multi-layer Structure

Results in [9] indicate that the DNN classifier has the highest accuracy when SNR is around 15 dB; therefore, we conduct simulations under this SNR and test the multi-layer RNN approach for up to 6 recursive layers. Note that we control the gain of all types of signals to be 15 dB, which means all the received signals in the database have the same SNR. We first show the impact of using different optimizers. The result are summarized in Figure 3, where we apply the multi-layer LSTM networks with different layers as an example. An SGD optimizer exhibits a better performance in a shallower network. It outperforms the ADAM optimizer with $\epsilon = 0.01$ when the number of layer is less than three. However, as more layers are integrated, the SGD's accuracy drops monotonously. In contrast, the ADAM optimizer with $\epsilon = 0.01$ has low accuracy in shallow architecture, but its performance improves when the network structure becomes deeper. The accuracy stabilizes at 64% with some fluctuations when the layer number is greater than three. An ADAM optimizer with $\epsilon = 0.001$ exhibits the best classification performance among these three optimizers. For instance, with three RNN layers, the classification accuracy enhances from 52.7% under SGD to 90.8% under ADAM . This is because an ADAM optimizer can individually adapt the learning step size for different θ by estimating of the first and second moments of the gradients. ADAM with $\epsilon = 0.001$ suffers slightly when the number of layer increases from three to six, but still outperforms other optimizers. As shown in Figure 3, The ADAM optimizer with $\epsilon = 0.001$ has a similar accuracy with $\epsilon = 0.01$ when the layer number is six. These results also indicate that even with the same DNN structure, different optimizer settings still have an impact on the performance of the neural networks.

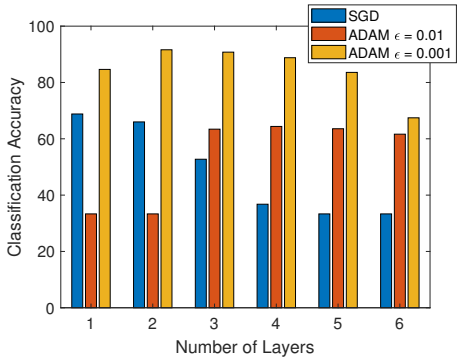


Fig. 3. Accuracy of multi-layer LSTM networks using different gradient optimizers.

TABLE I
COMPARISON OF RNN MODELS WITH VARIOUS NUMBER OF LAYERS.

Model	Number of RNN Layers					
	1	2	3	4	5	6
SimpleRNN	51.8%	70.8%	70.3%	73.6%	70.9%	68.4%
LSTM	84.6%	91.6%	90.8%	88.7%	83.6%	67.5%
GRU	85.5%	89.9%	90.4%	83.8%	66.4%	66.2%

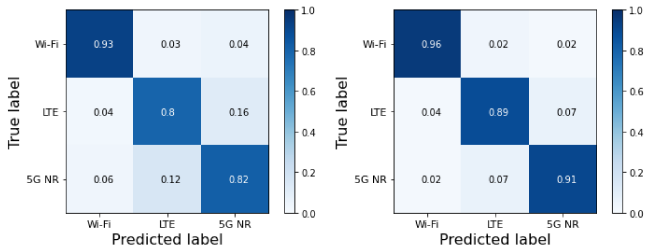
TABLE II
PRECISION AND RECALL FOR CLASSIFIED PROTOCOLS USING DIFFERENT RNNS UNDER SNR=20 DECIBEL.

Metrics	Protocol	GRU	Bi-GRU	LSTM	Bi-LSTM
Recall	Wi-Fi	82.00%	91.49%	83.34%	90.63%
	LTE	81.73%	90.19%	81.32%	89.34%
	5G NR	86.46%	85.58%	85.60%	91.09%
Precision	Wi-Fi	81.99%	86.00%	71.36%	88.00%
	LTE	85.00%	91.99%	90.91%	93.09%
	5G NR	83.01%	89.00%	88.18%	89.96%

Because the ADAM optimizer with $\epsilon = 0.001$ outperforms other optimizers, we apply it to the rest of RNN structures and show the results in Table I. It can be observed that the accuracy increases with the layer numbers for the first several layers but decreases when more layers are added. Among all these RNN-based classifiers, the LSTM network can achieve the highest accuracy of 91.6% when there are two LSTM layers stacked together. However, suffering from the layer increasing, the accuracy of the LSTM network drops to 83.6% when the layer number is five and to 67.5% when the layer number is six. This can be explained by the fact that the backpropagation of the gradient in deeper RNNs suffers more from exploding and vanishing problems than shallow RNNs. The redundant layers can also easily lead the model to overfit the training data and to perform worse on the testing part. We also tested different combinations of different types of RNN, and the results show similar trends (e.g., the accuracy increases for the first several layers and then decreases). These findings indicate that stacking RNN layers can improve the classifier performance but with certain limitations of the layer number.

C. Impact of Bidirectional Structure

We conduct simulations using the same dataset as before but with SNR = 20 dB. To calculate the recall and precision



(a) LSTM network. (b) Proposed LSTM network.

Fig. 4. Confusion matrix comparison between the single-layer unidirectional LSTM network and the multi-layer bidirectional LSTM network without recurrent-weight constraints.

for each label, we first define the true positive (TP) as the correct classification of the signal into such a label and the true negative (TN) as classifying other signal types into other labels. Accordingly, we also define two kinds of errors for the classifier: The false positive (FP) when misclassifying other types of signals into this label; the false negative (FN) when misclassifying this signal into other labels. Therefore, the precision can be written as $\frac{\text{TP}}{\text{TP+FP}}$ and the recall can be presented as $\frac{\text{TP}}{\text{TP+FN}}$. We then test the recall and precision of the proposed bidirectional structure in the LSTM and GRU as summarized in Table II. As shown above, the bidirectional architecture successfully improves the classification performance for both RNNs. The average recall improvement is 5.69% for GRU and 6.94% for LSTM, which indicates the bidirectional structure is more sensitive to detect FN samples. Meanwhile, the average precision improves 5.66% for GRU and 6.87% for LSTM. Thus, it shows that the enhancement of recall does not sacrifice precision performance. In other words, the bidirectional structure can reduce two types of errors (FP and FN) simultaneously. This improvement is because the proposed structure can benefit from the two-direction layers and help the RNN include the waveform dependencies from both sides to improve the classification performance. As a result, we consider combining the multi-layer structure with the bidirectional design.

The confusion matrix for the proposed combined neural network is depicted in Figure 4, where we use a three-layer bidirectional LSTM network as the example. As shown in Figure 4(a), the average accuracy for a single-layer unidirectional LSTM network is about 85%. It can detect the Wi-Fi signals more accurately compared with LTE and 5G NR signals. This is because the initial 5G NR launches depend on existing LTE infrastructure in non-standalone (NSA) mode and is similar to LTE protocol in 3GPP standardization. By introducing the proposed structure, the average accuracy can achieve 92%, as presented in Figure 4(b). Though LTE and 5G NR signals show lower accuracy compared with the Wi-Fi signals, the overall misdetections distribute more evenly in the proposed model. As a result, the proposed multi-layer and bidirectional LSTM network have a better classification performance than the basic LSTM network.

V. Conclusion

In this paper, we developed RNN-based deep neural networks to detect coexisting signal types by I/Q samples without having to decode them. With segmented sample sequences, different types of recurrent neural networks were trained. The classification result shows competitive accuracies by LSTM and GRU networks. We then applied the multi-layer and bidirectional structure to help capture long-term dependencies in the signals. However, the increasing complexity of the RNN can result in the over-fitting problem, so we proposed further improvements to compensate for it, including regularization and dropout, recurrent weight constraints, and learning rate halving. The classification accuracy gets further enhanced by the proposed structure. These results show that the proposed deep neural architecture can achieve accurate results in the signal protocol classification problems.

VI. Acknowledgements

This research was supported by the U.S. Army Small Business Innovation Research Program Office and the Army Research Office under Contract No. W911NF-21-C-0016, by NSF (grants CNS-1563655, CNS-1731164, and IIP-1822071), and by the Broadband Wireless Access & Applications Center (BWAC). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of NSF or ARO.

References

[1] "IEEE standard for definitions and concepts for dynamic spectrum access: terminology relating to emerging wireless networks, system functionality, and spectrum management," *IEEE Std 1900.1-2019 (Revision of IEEE Std 1900.1-2008)* , vol., no., pp.1-78, 23 April 2019.

[2] C. Xin and M. Song, "Analysis of the on-demand spectrum access architecture for CBRS cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 2, pp. 970-978, Feb. 2020.

[3] 3GPP, "Study on licensed-assisted access using LTE," *3GPP Work Item Description*, RP-141664, Sep. 2014.

[4] G. Naik, J. -M. Park, J. Ashdown and W. Lehr, "Next generation Wi-Fi and 5G NR-U in the 6 GHz bands: Opportunities and challenges," *IEEE Access*, vol. 8, pp. 153027-153056, 2020.

[5] W. C. Headley and C. R. C. M. d. Silva, "Asynchronous classification of digital amplitude-phase modulated signals in flat-fading channels," *IEEE Transactions on Communications*, vol. 59, no. 1, pp. 7-12, January 2011.

[6] M. Hirzallah, W. Afifi and M. Krunz, "Full-duplex-based rate/mode adaptation strategies for Wi-Fi/LTE-U coexistence: a POMDP approach," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 1, pp. 20-29, Jan. 2017

[7] W. Zhang, M. Feng, M. Krunz and A. Hossein Yazdani Abyaneh, "Signal detection and classification in shared spectrum: A deep learning approach," in *Proc. IEEE INFOCOM conference*, May 2021 pp. 1-10

[8] 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.

[9] T. J. O'Shea, J. Corgan, and T. C. Clancy," Convolutional radio modulation recognition networks," in *Proc. International conference on engineering applications of neural networks*, 2016, pp. 213-226, Springer.

[10] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," in *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433-445, Sep. 2018

[11] W. Zhang, M. Krunz, and G. Ditzler, "Intelligent jamming of deep neural network based signal classification for shared spectrum," in *Proc. IEEE MILCOM Conference*, San Diego, Nov. 29 – Dec. 2, 2021.