IEEE Open Journal of
Circuits and Systems

Received 22 February 2025; revised 5 May 2025 and 10 July 2025; accepted 21 July 2025. Date of current version 18 August 2025.

Digital Object Identifier 10.1109/OJCAS.2025.3592376

Prediction-Based Spectrum Sensing Framework for Cognitive Radio

ANDRES ROJAS[1,2] (Graduate Student Member, IEEE), GAWEN FOLLET[3], GORDANA JOVANOVIC DOLECEK[1] (Life Senior Member, IEEE), JOSÉ M. DE LA ROSA[2] (Fellow, IEEE), AND GUSTAVO LIÑÁN-CEMBRANO[2]

[1]National Institute of Astrophysics, Optics, and Electronics, 72000 Puebla, Mexico
[2]Instituto de Microelectrónica de Sevilla, IMSE-CNM (CSIC/Universidad de Sevilla), 41092 Sevilla, Spain
[3]Polytech Montpellier, 34095 Montpellier, France

This article was recommended by Associate Editor R. Colella.

CORRESPONDING AUTHOR: JOSÉ M. DE LA ROSA (e-mail: jrosa@us.es)

This work was supported in part by Secretaría de Ciencia, Humanidades, Tecnología e Innovación (Secihti) and by Grants PID2022-138078OB-I00, and PDC2023-145808-I00, funded by MICIU/AEI/10.13039/501100011033, by Grant USECHIP (TSI-069100-2023-001), Project funded by the Secretary of State for Telecommunications and Digital Infrastructure, Ministry for Digital Transformation and Civil Service and by the European Union NextGenerationEU/PRTR, by ERDF A way of making Europe.

ABSTRACT This paper presents a hardware-software deep learning architecture for prediction-based spectrum sensing in Cognitive Radio (CR) applications. A convolutional neural network-based predictor for spectrum occupancy was trained using the band power from I/Q samples acquired by a software-defined radio (SDR). Additionally, a second neural engine was trained for radio frequency (RF) frame detection based on spectrograms. We implemented a transfer-learning solution using a You-Only-Look-Once version 8 nano model with a synthetic dataset comprising thousands of wireless signals, including Wi-Fi, Bluetooth, and collision frames. Once trained, the two neural networks were transferred to a Raspberry Pi 5 – an affordable single-board computer – connected to two (one for Rx, one for Tx) ADALM-PLUTO SDR systems for benchmarking using over-the-air signals in the 2.4 GHz band. Together with our methodology and experimental results, the paper also presents an overview of current spectrum prediction proposals and RF frame detectors. Remarkably, to the best of our knowledge, this proposed framework is the first approach towards an Internet of Things-suitable implementation of prediction-based spectrum sensing for CR applications.

INDEX TERMS Cognitive radio, deep learning, spectrogram, spectrum occupancy prediction, spectrum sensing, software defined radio.

I. INTRODUCTION

BILLIONS of smart devices and emerging applications such as autonomous vehicles, robotics, augmented reality, etc., are seamlessly connecting to the Internet, creating a rapidly growing number of bandwidth-hungry mobile broadband devices [1]. Users demand more spectrum resources and better Quality of Service (QoS), especially for latency-sensitive applications [1]. For this reason, radio frequency (RF) signal detection and classification are critical in wireless systems, especially for cognitive radio applications [2]. This paper presents an extension of our proposal in [3] for a deep learning (DL)-based architecture for RF frame detection. The current work implements an Artificial-Intelligence (AI) based solution for prediction-based spectrum sensing, which has been widely investigated in cognitive radio communications research to increase spectral efficiency.

A. COGNITIVE RADIO AND SPECTRUM SENSING

Cognitive radio and software-defined radio are highly related concepts [4]. CR uses SDR to implement generic hardware, intending to replace more specific ones [5]. This feature allows CR users to be aware and adaptable to the radio environment, exploiting the underused resources while avoiding harmful interference with primary user (PU)

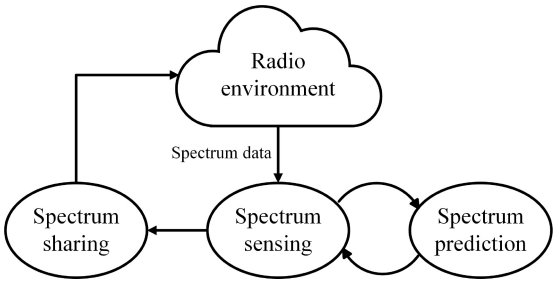

FIGURE 1. The working process of prediction-based spectrum sensing in cognitive radio networks.

communications [6]. Some CR networks' key technologies include spectrum sensing (SS), spectrum sharing, and spectrum prediction [7]. Fig. 1 illustrates the operation sequence of prediction-based SS for CR users in a dynamic spectrum access framework. First, SS obtains the spectrum state by detecting the free frequency bands or "spectrum holes" from the radio environment. Secondly, the spectrum prediction uses historical SS information to discover occupancy patterns, creating a prediction of spectrum occupancy in future time slots [7]. Spectrum sharing receives the prediction and decides the allocation for transmitting information from the CR user. CR divides spectral users into licensed users or PUs and unlicensed or secondary users (SU). PUs can access the spectrum whenever they need it, in contrast, SUs must be aware of PUs to avoid interference [6]. The necessity of sharing the spectrum comes from the overuse of several frequency bands, such as the 2.4 GHz industrial, scientific, and medical (ISM) band, which is why SUs must acknowledge PU signals using SS. The interweave paradigm of CR allows SUs to transmit using their maximum power only when the PU is absent [6], [8]. This paradigm is known as the classical CR and it was implemented in this paper using the half-duplex approach known as listen before talk (LBT). The LBT protocol states that the SU performs SS periodically in two slots: sensing and transmission [6]. During the sensing slot, the SU silently performs SS avoiding interfering with its decision reliability. For this reason, LBT is also known as half-duplex cognitive radio (HDCR). When the PU is absent, the SU changes to the transmission slot and transmits data without being aware of the PU activity [6]. The LBT approach was implemented in this work.

B. SPECTRUM OCCUPANCY PREDICTION

Spectrum occupancy prediction provides agility and adaptability to CR functions optimizing the periodic SS scheduling and channel selection [9]. In other words, occupancy prediction for SS allows the CR users to skip the sensing duty on some channels predicted to be busy reducing the sensing time and energy consumption [10]. Prediction-based SS predicts the requests from CR users to pre-assign certain bands for effective spectrum sharing before the CR requests arrive. This approach can better exploit the channel capacity and reduce the response delay [10].

C. OBJECT DETECTORS

In recent years, the research community has proposed SS approaches based on data-driven detectors leveraging techniques originally aimed at computer vision. These methods improve the performance and usefulness by adopting data-centric strategies for sensing algorithms [11]. One of the benefits of using this approach is achieving sensing systems that generalize well, are resilient to impairments, and accomplish good sensing performance in a wide range of scenarios. Signal detection and classification from spectrograms are analogous problems to object detection in computer vision. Indeed, human visual performance in this task is quite good when considering manual decisions made while looking at a spectrogram, which is the feature representation that we have employed in this proposal [3], [12]. Since operation speed is important for radio emission detection and classification we relied on the popular You Only Look Once (YOLO) [13] framework, a Convolutional Neural Network (CNN)-based object detection architecture providing an excellent balance between detection accuracy and processing time [14], thanks to its operation in a single forward pass.

D. CONTRIBUTIONS, NOVELTY, AND ORGANIZATION

In this article, we start with an overview of recent DL-based works for CR applications and then propose a prediction-based spectrum sensing architecture. We describe the methodology and present experimental results using real-world spectrum data captured using the ADALM-PLUTO SDR from Analog Devices [15]. Finally, we discuss these results and compare them with recent spectrum predictors from the literature. Our main contributions are detailed as follows:

- We survey recent works for CR applications based on DL techniques for RF frame detection and spectrum occupancy prediction. In particular, we first discuss works implemented for RF frame detection using spectrograms and object detectors. A detailed summary table is included to compare various approaches and show the recent progress in this field. Besides, spectrum occupancy prediction works were also surveyed. By studying the variety of approaches in the state of the art, we divided this overview into spectrogram-based and time series-based approaches.
- We propose a DL-based architecture for prediction-based spectrum sensing. This system combines a CNN-based spectrum occupancy prediction and a YOLOv8-based [12] RF frame detector [3]. The software implementation was performed using multi-threading over the 4 cores of the Cortex A-64 processor in the Raspberry Pi 5, an affordable single-board computer (SBC).
- We compare our proposal against recent spectrum prediction approaches from the literature, showing that this is the first Internet of Things (IoT) suitable implementation of prediction-based SS. This framework

is a step forward in the hardware implementation of artificial intelligence (AI)-based IoT devices.

The novelty of this work includes the following:

- We introduce an IoT-suitable CR implementation for spectrum prediction, supported by multiple experimental performance evaluations.
- We describe a DL-based approach for prediction-based spectrum sensing using time signals and spectrograms. Combining both techniques provides more control for spectrum sharing.
- We present a proof of concept system built with two affordable SDRs (ADALM-PLUTO) and an SBC (Raspberry Pi 5), which has been tested using real-world signals.

II. OVERVIEW OF DEEP LEARNING-BASED WORKS FOR CR APPLICATIONS

The state of the art on SS and spectrum occupancy prediction covers a variety of proposals. We divide this overview into RF frame detection and spectrum occupancy prediction.

A. RF FRAME DETECTION

This section deepens into an overview of recent RF frame detection works relying on object detector-based approaches using spectrograms. The analogy to an image processing task is straightforward; spectrograms are bi-dimensional (frequency vs. time) data arrangements whose values correspond to the average power of RF signals in the frequency interval and temporal window associated with each particular coordinate (pixel position). Hence, one can simply consider spectrograms as images from where the "objects," i.e., wireless signals, need to be detected.

A wideband SS technique to detect and localize wireless RF signals in time and frequency was reported in [16]. This approach adopted and downscaled the faster region-based CNN framework extracting features via the VGG-13 architecture [17]. The authors proved that a better mean average precision (mAP) could be achieved by using a mixture of synthetic and real RF traces in the training. Another DL-based network in [2] merged RF detection, classification, and localization. The authors used a YOLOv4 detector that leveraged RF-centric compression and enhancement over the input spectrograms providing better mAP for a wider signal-to-noise ratio (SNR) range. In [18], the authors proposed a two-stage spectrogram-based coarse and fine method to detect and characterize radar pulses in high-noise environments. This proposal employed two different YOLOv3 frameworks, the first was trained on 100 MHz-extent spectrograms for coarse signal detection and the other received cropped and stretched signals for a finer detection. Similarly, [19] presented a two-stage drone signal detection project based on spectrum localization and YOLO-based classification. This technique combined goodness-of-fit sensing for detection and a deep residual neural framework for classification. This article proved that YOLO provides better inference time than deep neural network approaches. The work in [20] proposed a YOLOv2-based hybrid architecture for detecting narrowband IoT-related technologies including LoRa and Sigfox. Their proposal applied image processing techniques to create spectrograms based on real measurements.

Another YOLO-based detector in [21] for cooperative SS described a time-frequency localization framework. This lightweight object detector implemented a fusion algorithm based on non-maximum suppression from SU information. Likewise, a consistency-constrained dual generative adversarial network (CCD-GAN) for domain adaptation was proposed in [22]. This methodology enforced time-frequency localization consistency using features such as signal correlation, signal power, texture, and shape in the spectrogram. The system addressed the consistency of the source spectrum and the target spectrum (real samples). The first practical implementation of reconfigurable intelligent surfaces (RIS) for SS was presented in [23]. The system employed a real RIS-empowered DL prototype where wireless communication between the PU and SU is assumed to occur through the RIS's reflected signals. A YOLOv7 model was trained using the spectrograms of a synthesized dataset including 4G long-term evolution (LTE) and 5G new radio (NR) signals. These results proved that YOLOv7 detected 4G LTE and 5G NR signals more accurately.

Another RF signal detection system in [24] proposed an RF region-of-interest CNN (RFROI-CNN) solution for wideband SS. This method applied CNNs and expanded the energy detection approach by estimating the noise distribution in the spectrogram. This approach detected significantly weaker signals than traditional thresholding methods. The ROI accounted for neighboring signal power values in the analyzed spectrogram segment, and finally, the model detected signals by subtracting the noise estimate from the input spectrogram. The YOLO-based method reported in [25] addressed wideband signal recognition using YOLOv5 for detecting, localizing, and classifying narrowband signals combined with a CNN-based model for automatic classification of modulation schemes. The agile IoT network communication platform for SS in [26] uses a combination of energy detection and semantic spectrum segmentation. This DL approach reduces the computation complexity, but relies on powerful computation resources such as GPUs.

The latest work for RF frame detection in [3] proposed a YOLOv8-based approach for classifying Wi-Fi and Bluetooth (BT) signals. This proposal implemented this architecture on a Raspberry Pi 5, showing the benefits of this object detector for simpler devices and compact DL-based systems dedicated to CR applications.

Table 1 summarizes recent works for RF identification using DL-based object detectors and spectrograms. In terms of wireless signals, there is a wide variety in the literature including LTE and 5G [18], [21], [22], [23], LoRa and

TABLE 1. Summary of recent works for RF emission identification.

Article	Dataset		Number of spectrograms	Detector	Complexity					
	Technologies	Classes			Inference time	Prediction time	Total parameters	Processor		
								Training	Testing	IoT suitable?
[16]	Wi-Fi, BT, Microwave	3	200	Downscaled FRCNN	N/A	N/A	N/A	N/A	N/A	N/A
[2]	Wi-Fi, BT, ZigBee, XPD, Lightbridge	5	99330	YOLOv4	22.96 ms	N/A	N/A	GTX 1080	GTX 1080	No
[18]	Radar, LTE, 5G	3	1300	YOLOv3	N/A	N/A	N/A	N/A	N/A	N/A
[19]	Drone, Wi-Fi	10	7000	YOLO-Lite	1.16 s	0.552 ms	41.53M	RTX 2080	RTX 2080	No
[20]	LoRa, Sigfox	7	19320	YOLOv2	N/A	N/A	N/A	RTX 2070	RTX 2070	No
[21]	LTE, 5G	2	N/A	YOLOv1	N/A	N/A	0.82M	N/A	N/A	N/A
[22]	LTE, 5G	2	N/A	YOLOv1	N/A	N/A	11.38M	GTX 1080	GTX 1080	No
[23]	LTE, 5G	2	2700	YOLOv7	N/A	N/A	N/A	N/A	N/A	N/A
[24]	AM, FM, CW, LSB, USB, OFDM	6	5189	RFROI-CNN	65.31 ms	21.75 ms	N/A	RTX 3060	i7-11800H	No
[25]	PAM, FSK, AM, GMSK, FM, OFDM	12	10900	YOLOv5	N/A	N/A	9.07M	RTX 2080	RTX 2080	No
[26]	BPSK, QPSK, 16QAM, PAM4, GFSK, OQPSK	6	N/A	DNN	N/A	N/A	0.46M	RTX A4000	i7-12700K	No
[3]	Wi-Fi, BT, Collision	3	20000	YOLOv8	18.33 ms	0.208 ms	3M	RTX 3060	ARM Cortex A76	Yes

Sigfox [20], Wi-Fi and BT [2], [3], [16], [19]. Table 1 also compares the mean inference time, defined as the time to complete a full cycle using the proposed system, although this result is only reported in a few works [2], [3], [19], [24] it is crucial to emphasize that only [3], [24] and [26] employed a CPU-based processor for testing (in contrast with works testing over GPUs). The mean prediction time is the execution time for producing an output utilizing the object detector. The shortest time was obtained using YOLOv8 as reported in [3]. Another complexity parameter compared in Table 1 is the number of model parameters. The two smallest values correspond to [26] and [21], which applied energy detection-based and cooperative approaches, respectively, but neither of them reported their inference times nor their prediction times. Other values include 3 million for [3]; much smaller than the approaches in [19], [22] and [25]. Finally, the last column in Table 1 indicates whether the proposal is suitable for IoT devices. We categorize the literature by comparing the testing processor used during their experiments. It is noticeable that only the work presented by the authors of this paper in [3] used a processor for IoT devices. The rest of the works either did not report this information or employed powerful GPUs.

B. SPECTRUM OCCUPANCY PREDICTION

Several approaches for spectrum prediction have been proposed in recent years. For the sake of clarity, these works are divided into spectrogram-based and temporal series-based approaches.

1) SPECTROGRAM-BASED APPROACHES

The temporal-frequency fusion attention approach in [27] predicted heterogeneous data energy levels obtaining intra-spectrum correlations from temporal and frequency domains through attention mechanisms. A fusion module combined this information, and a Long Short-Term Memory (LSTM) model implemented the prediction. A combination of CNNs and Gated Recurrent Unit (GRU) proposed by [28] leveraged the local and regional correlations from spectrum data. This proposal combined 1D and 2D-CNN to find correlation patterns. The authors implemented a GRU because it has fewer training parameters than LSTMs. In contrast, a 2D-LSTM-based proposal in [29] predicted spectrum opportunities using input binary occupation grids and produced a binary output as the predicted occupancy. The authors used real-world spectrum measurements as the dataset for training. Similarly, the spatiotemporal prediction model in [30]

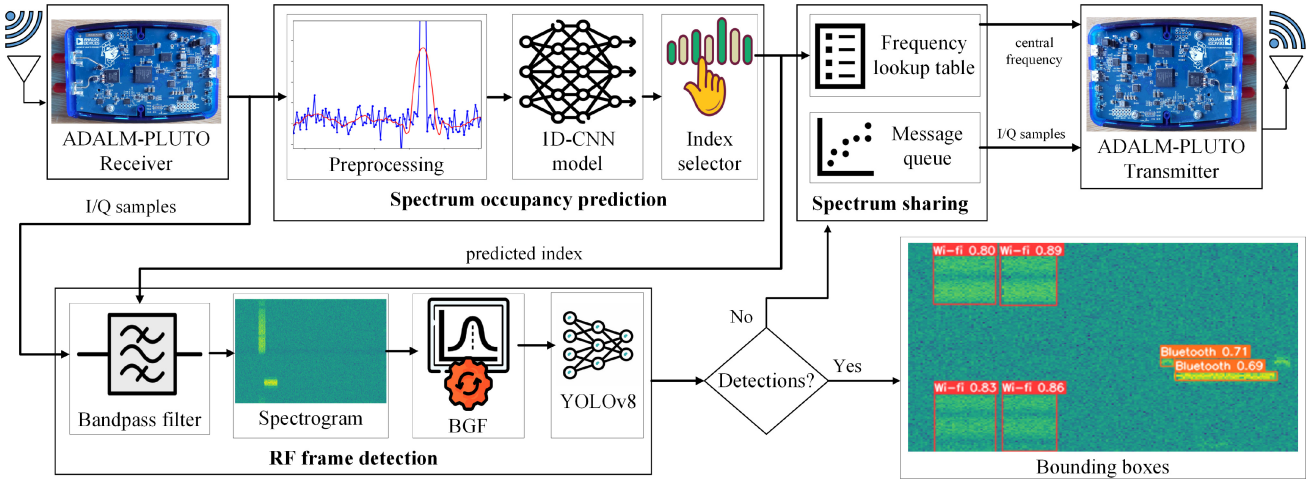

FIGURE 2. Global architecture and conceptual signal flow in the proposed system.

implemented a combination of CNNs and residual networks (ResNet). The former captured spatial and temporal features, while the latter avoided the vanishing gradient problem. This dual technique ensured stable weight updating during the back-propagation process. Another combination between CNN and LSTM in [31] predicted future sequences from frequency-hopping signals. This two-stage method applied YOLO for signal detection and LSTM for time-frequency prediction. The proposal achieved high accuracy while reducing computational costs compared with deeper models such as Convolutional LSTM (ConvLSTM) or Bidirectional LSTM (BiLSTM). Likewise, the deep network proposed in [7] used Bi-ConvLSTM combined with attention-based mechanisms. The system received frequency-time grids from spectrum states measured in a residential area (real-world data). The authors compared their proposal against several networks such as LSTM, GRU, BiLSTM, CNN, etc.

2) TEMPORAL SERIES-BASED APPROACHES

The proposal in [32] predicted the average received power using an autoregressive-moving-average (ARMA) model combined with a low-pass filter. This work compared the ARMA and LSTM approaches showing that they had similar accuracy with preprocessed data. In contrast, the authors of [33] designed a hierarchical spectrum prediction model based on GRU offering an accurate prediction on the spectrum availability for the SU considering the prior information of PU. Performance was illustrated through real spectrum measurements using the minimum and maximum recorded data for training and prediction. Another spectrum prediction system reported in [34] introduced an LSTM-based selector to find the optimal window of future forecasts. This selection would increase the network utilization while reducing the interference caused by SUs. The LSTM model was reliably used to guide the channel selection for the opportunistic user. Similarly, a framework for spectral transmission based on the LSTM prediction was reported in [35]. The prediction and sensing-based spectrum access system employed an encoder-decoder LSTM to predict two time slots simultaneously and employed the prediction result to improve the SU's waiting time and capacity. A comparison of several Artificial Neural Networks (ANN) for spectrum prediction was reported in [36] including 1D-CNN, LSTM, autoencoder LSTM, and CNN-LSTM. These models were trained with channel occupancy temporal series. Synthetic and experimental data were used to assess the computation time and accuracy performance. Another work reported in [37] proposed a dual branch neural network mobile spectrum prediction model (DB-LSTM) combining LSTM and BiLSTM. The SS stage extracted intrinsic signal features using a ResNet module and a CNN. These characteristics were the input to train the DB-LSTM model for spectrum prediction. Finally, a recent work was proposed in [38] employing a variational mode decomposition as preprocessing to reduce the high data volatility. After this, an LSTM was trained in combination with a dual-attention mechanism. The authors devoted this proposal to multi-band spectrum prediction tasks on real spectrum datasets.

III. PROPOSED ARCHITECTURE FOR PREDICTION-BASED SPECTRUM SENSING

The work in this paper aims to predict spectrum occupancy values for spectrum sensing in cognitive radio applications. Our solution, whose overall architecture is illustrated in Fig. 2, involves three main operations, namely: spectrum occupation prediction, RF frame detection, and spectrum sharing.

A. SPECTRUM OCCUPANCY PREDICTION

The spectrum occupation prediction block comprises four main tasks: SDR sample acquisition, preprocessing, CNN-based spectrum occupancy prediction, and channel index selection for transmission. Below is a detailed explanation of these steps.

1) SDR SAMPLE ACQUISITION

RF signal acquisition is implemented with the ADALM-PLUTO SDR [15], a single-input-single-output (SISO) SDR module[1] with a [325 MHz, 3.8 GHz] RF signal tunning range, [200 kHz, 20 MHz] tunable channel bandwidth, integrated 12-bit DACs (Tx) and ADCs (Rx), and [65.1 kSPS, 61.44 MSPS] variable output data rates. In our setup, the ADALM-PLUTO board is connected to the Raspberry Pi 5 through a USB connection. The user defines the central frequency of the wideband signal to be captured. The interface between the SDR and Python is easily managed by means of the Analog Devices' *adi* API. The default SDR configuration parameters are the center frequency, set to 2.412 GHz, the sample rate, set to 60 MHz, and the number of samples per frame, set to 5120. In any case, the user can always modify (within allowed ranges) these values to optimize them for a particular application.

It is important to clarify that, in this work, the center frequency of the sensed wideband signal is user-defined. In practice, CR systems are expected to autonomously identify and operate on the least congested frequency slot within a given spectrum range. However, it is always necessary to establish a specific spectrum region where the SS system will search for and locate such a frequency slot. For the rest of this paper, we use channel 1 from the 2.4 GHz ISM band (2.412 GHz) as the center frequency for our system.

2) PREPROCESSING

The quadrature I/Q data captured by the SDR require certain preprocessing before they can be passed to the spectrum-prediction neural module. The algorithm needs the center frequency, bandwidth, and the number of allocable frequency slots; i.e., the number of divisions inside the sensed wideband. After getting the temporal signal (I/Q complex samples) and computing its Fast Fourier Transform (FFT) the data is split into the defined slots. The average band power per slot is computed and filtered to get a usable signal for the CNN. First, the signal's envelope is computed and then a low-pass filter is applied. This process creates an occupancy value for every slot which is finally min-max normalized to a [0, 100] interval [36]. Fig. 3 shows an example of occupancy levels computed from over-the-air signals captured by the ADALM-PLUTO SDR.

3) CNN-BASED SPECTRUM OCCUPANCY PREDICTION

1D-CNNs are a variant of traditional CNNs, specifically conceived to operate on one-dimensional sequential data. These models leverage the concept of convolution, which involves applying a set of learnable filters to input data to extract meaningful features. The key idea is to capture local patterns in the data by performing convolutions across different regions. In this case, we focus on the use of 1D CNNs for temporal evolution forecasting since these networks can effectively capture patterns in sequential data [39]. The structure of our prediction model comprises the following layers:

[1]Data from: https://wiki.analog.com/university/tools/pluto/devs/specs.

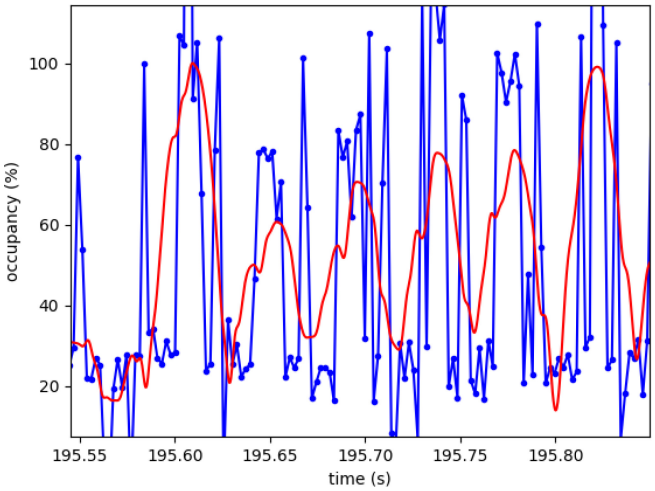

FIGURE 3. Example of real occupancy levels. The blue line displays the unfiltered band power whereas the red line shows the final filtered result.

1) Input layer: Composed of 1D-tensors corresponding to the number of slots.
2) Convolutional layer: This layer applies convolution operations to extract feature maps using the tensors from the previous layer. The layer contains 64 trainable kernels (size = 3).
3) Pooling layer: Receives the feature maps from the convolution layer and retrieves the most relevant features using max pooling of size 2.
4) Flatten layer: This layer reshapes the data from 2D-feature tensors back to 1D tensors since the network will produce a one-dimensional prediction.
5) Dense layer: A fully connected layer that interprets the flattened features using weighted operations between multiple neurons.
6) Output layer: A second dense layer that receives a map of features from the previous layer and generates the prediction.

For each frequency slot whose occupation is to be predicted, we create its 1D-CNN block with the layer configuration described above. The goal of the prediction unit is to identify the index (a particular frequency slot) that will have a lower occupation in the near future (in the order of 150ms). Fig. 4 shows a diagram of the data acquisition time considered when creating the input dataset based on real-world signals. The occupancy dataset was created with data gathered considering the computation time (t_c) for the band power processing and CNN inference. The time ahead (future length, l_f) we want our prediction to fit is the number of milliseconds we want to include as predictions for the neural module. Therefore, the forecast time (t_f) for spectrum occupancy prediction will be $t_c + l_f$. We set $t_c = 100$ ms and $l_f = 50$ ms to accommodate the Raspberry Pi 5 processor and produce useful predictions. These times were

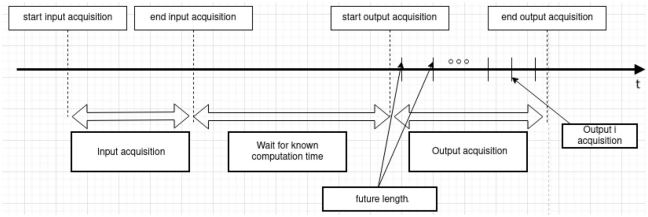

FIGURE 4. Data acquisition time for the spectrum occupancy prediction dataset.

found empirically. Using these values, our forecast time will be 150 ms meaning that at most our system can predict occupancy values 150 milliseconds in advance.

The training for the spectrum occupancy predictor was executed on an NVIDIA GeForce RTX 3060 Ti (CUDA version 11.5) using a total of 5000 examples with a specified number of input and output samples–though we also provided the functionality to let the user manually configure values. Each example was produced from the acquired I/Q samples using the pre-processing steps described previously.

The training process for each 1D-CNN block depends on the number of epochs established by the user. By default, our system uses 30 epochs and a batch size of 14. The training for the spectrum occupancy predictor, on average, takes 96 seconds. Fig. 5 shows an example of the spectrum occupancy prediction results.

4) INDEX SELECTION

As mentioned in the introduction, the goal of this work was not only to predict the occupancy of certain predefined frequency slots but also to select the best band–defined here as the least occupied–for transmission by the CR hardware. Since real systems can not switch between transmission frequency slots at arbitrary speeds, we implemented a solution that employs information from the previous occupancies, selected indexes, and four extra parameters, namely: hd – history depth, occ_{max} – normalized maximum occupancy value, th – a threshold (in %), and min_ts, – the minimum switching time. First, for each frequency slot, an occupation history vector O_k, with $k = [1, 2, \ldots, hd]$ elements is created with the last hd occupancy values. To avoid switching from one slot to another too fast, the algorithm checks if a switch has already been done during the last min_ts seconds. The algorithm selects a new index if the new value is $occ_{max} * th$ times smaller than that of the previously selected slot. Otherwise, it keeps the previous selection. The occ_{max} parameter is the maximal amplitude of the occupancy signal and th is a value from 0 to 1 representing a minimum occupancy change to be reached before switching.

B. RF FRAME DETECTOR

The RF frame detector implemented in this system is improved from our previous work in [3], and conveys the following operations.

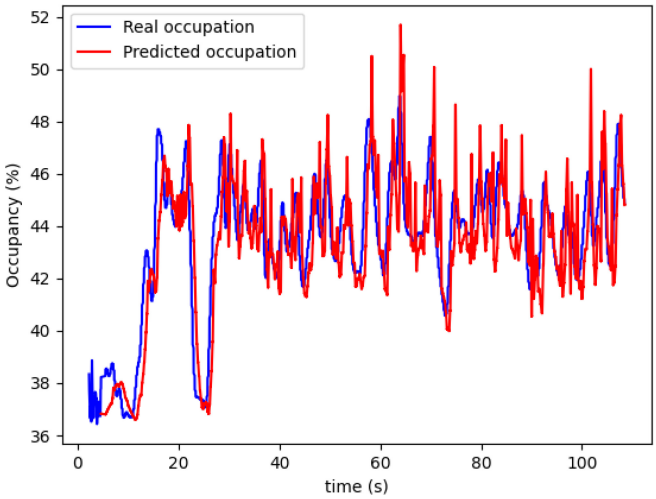

FIGURE 5. Example of real vs. predicted occupation.

1) BANDPASS FILTERING

Firstly, a bandpass filter, centered on the frequency slot provided by the index selection algorithm is introduced within the spectrogram computation thread. The filter is programmed to have a bandwidth corresponding to 6 slots. Since the SDR bandwidth was set to 60 MHz, each frequency slot width is simply given by $BW_s = (60/n_{slots})$ MHz. For example, when $n_{slots} = 60$, each slot bandwidth is $BW_s = 1$ MHz. Hence, if the central frequency associated with the selected index i is f_i, the filter will reject all signals except those between $[f_i - 3BW_s, f_i + 3BW_s]$.

2) SPECTROGRAM DATASET

The dataset employed for transfer learning using the YOLOv8n model was extracted from [40]. It contains 20000 images including a total of 362780 Wi-Fi frames, 21340 Bluetooth frames (Low Energy (BLE) 1 MHz, BLE 2 MHz, and Classic), and 77600 collisions between frames. In our implementation, the horizontal axis in the spectrogram corresponds to the time in milliseconds, the vertical axis corresponds to the frequency in MHz, and the pixel value represents the reception strength (dBm)–encoded using the viridis color map [41]. Besides the large number of samples available for training the neural model, an important reason for choosing this dataset lies in its generation pipeline, which is based on different RF standard frames produced by a Rohde & Schwarz SMBV100B vector signal generator that were recorded by a SDR [40]. The dataset also includes data augmentation, where individual frames are randomly arranged according to policies corresponding to the RF standards to form larger signal sections of 4.5 ms. Details regarding the parameters used to simulate these signals can be found in [40].

3) BILATERAL GAUSSIAN FILTER

The use of bilateral Gaussian filter (BGF) proposed in [8] and [42] to address spectrogram denoising for SS was

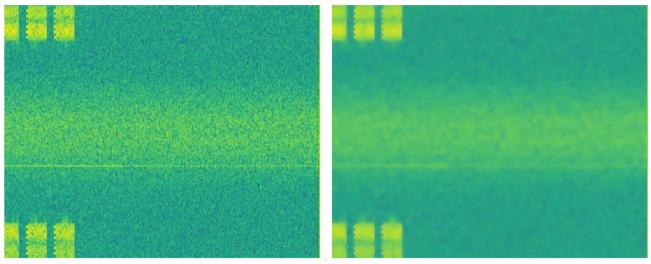

FIGURE 6. Examples of spectrogram denoising using the BGF. Left: original spectrogram. Right: denoised version.

included in our framework. The BGF smooths pixels using Gaussian kernels taking into account the closeness to edges. The key idea here is to enhance the visual separation between RF signals and the background noise in the spectrograms. This denoising image filter was applied to the spectrograms produced by the bandpass filter. Fig. 6 shows an example of a spectrogram before and after using the BGF.

4) YOLOV8N-BASED OBJECT DETECTOR

Transfer learning was applied to a YOLOv8n-based object detector using the spectrogram dataset [40] to detect Wi-Fi, Bluetooth, and Collision frames. YOLOv8 is a family of deep neural network solutions supporting tasks like object detection, instance segmentation, image classification, pose estimation, and multi-object tracking. For each task, different model variants offer optimized trade-offs between accuracy and complexity (numbers of parameters). In this work, the nano version of YOLOv8 was used for transfer learning using the spectrogram dataset. Fig. 7 presents the single-stage object detection framework based on YOLOv8n implemented in this work. The model includes the backbone, which computes features using convolutional operations over the input spectrograms, providing information to the classification and regression heads. The bounding box specifies the spatial coordinates (*x, y, w, h*) of the located object, and the object class identifies the label for the detected object. Three classes were included in this work; Wi-Fi, Bluetooth, and Collision.

The training process was implemented using the Ultralytics library [12] for Python on an NVIDIA GeForce RTX 3060 Ti (CUDA version 11.5) for 300 epochs, setting the batch size to 128. The 20000 samples in the dataset were randomly divided into training (14000), validation (4000), and test sets (2000) –i.e., (70%, 20%, 10%). The training process for YOLOv8n takes about 9 hours, which is much longer compared to the 1D-CNN block for spectrum occupancy prediction, since we are dealing with a larger dataset of 2D signals. However, as with any other DL-based system, the training process was performed only once, and the inference over real data is much faster (0.208 ms on average [3]). Accuracy was determined according to two commonly employed metrics in object detectors: the mAP50 and the mAP50-95. Both metrics evaluate the similarity between the detected box and its corresponding ground

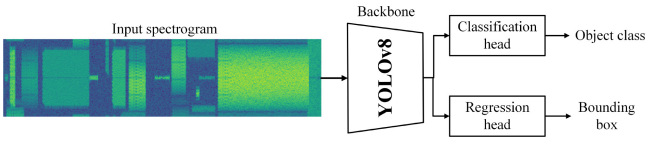

FIGURE 7. Single stage object detection using YOLOv8n and spectrograms from [40].

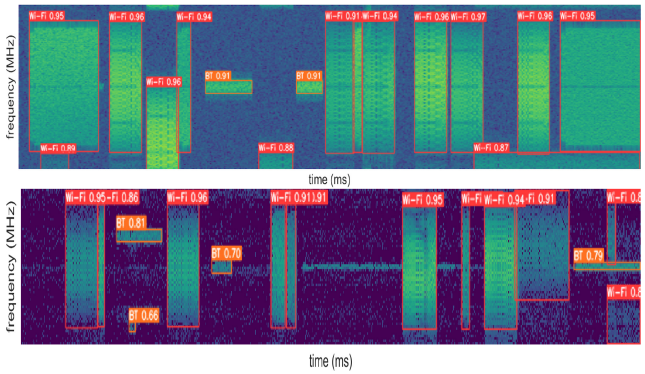

FIGURE 8. Detection examples over two spectrograms from the test set [3].

TABLE 2. Validation results for RF frame detection [3].

Class	Precision(%)	Recall(%)	mAP50(%)	mAP50-95(%)
All	89.6	62	71.7	61.3
Wi-Fi	95.6	94.7	97.2	92.6
Collision	79.7	53.3	58.6	47.8
Bluetooth	93.4	38	59.4	43.4

truth. Boxes are compared using the Intersection over Union metric (IoU). Specifically, mAP50 provides the mean average precision at an IoU threshold of 50%, while mAP50-95 extends the evaluation to 95%. The results for the validation dataset are provided in Table 2. The RF frame detector provides the highest mAP50 (97.2%) for the Wi-Fi class, which is not surprising though, since class density in the dataset is biased towards Wi-Fi frames. Fig. 8 shows the detection results for two spectrograms from the test set with their confidence scores. It can be noticed that the Wi-Fi and Bluetooth signals are detected with confidence above 0.8. The different background colors in the two cases are due to different noise levels.

C. SPECTRUM SHARING

With the development of AI techniques, important efforts have been made to develop intelligent spectrum-sharing solutions. AI-enhanced spectrum sharing is a good solution because improves spectral efficiency by reducing the interference and enabling sharing in multiple domains such as time, frequency/band, or spatial [1]. In this proposal, the spectrum-sharing module uses the index selection output and the bounding boxes from the RF frame detector to decide whether to transmit or not a test signal modulated in the predicted frequency slot.

Regarding incumbent users, those with legacy access rights to the spectrum (generally, federal or satellite systems) [43], the current proposal uses a general signal classification, accounting for every RF frame detected as a PU or an incumbent user. Incumbent protection/awareness is a major challenge because incumbent users are highly mobile nodes that dynamically utilize the spectrum in spatial and temporal dimensions [43]. In our case, since we used indoor data with real occupancy values, we included a wide variety of real transmissions present within our laboratory, which inherently will include incumbent user data. Finally, our spectrum-sharing module uses a -30 dB gain for transmission. This value was established empirically according to our indoor environment. By limiting the transmission power of the SU, we avoid causing interference with incumbent users or PUs [44]. Each time an RF frame is detected, our system immediately releases the band by clearing the transmission buffer used by the ADALM-PLUTO SDR.

The RF frame detector acts as a safeguard for unnecessary channel vacating. This approach was originally devoted solely to spectrum sensing, but with the introduction of the spectrum sharing block, we must account for realistic noise and interference, as well as false positives. The spectrogram dataset [40] used to train the YOLOv8 nano model includes the Collision class. This "type" of signal considers the case of interfering frames under realistic conditions. We can look back at Table 2 and notice that the precision validation result is 89.6% for all the classes. This means that the RF frame detector detects correctly almost 9 out of 10 frames. We can also see from Table 2 that the precision for Wi-Fi and Bluetooth is 95.6% and 93.4%, respectively. These are high indicators of true positive detections. Accounting for false positive mitigation, we would improve our proposal by increasing the spectrogram dataset with more examples, including real over-the-air signals generated in a controlled environment. Of course, there are also other algorithm-based approaches to mitigate false detections like the reputation-based mechanism in [45] or the hidden Markov-based weighted fusion scheme in [46]. In future work, we would explore these approaches to improve our current prototype.

IV. IMPLEMENTATION AND EXPERIMENTAL RESULTS

A. RASPBERRY PI IMPLEMENTATION

The proposed deep learning-based framework was implemented on a Raspberry Pi 5 to verify its behavior on over-the-air RF measurements. Fig. 9 shows the three devices used in this system, namely; two ADALM-PLUTO SDRs for RF signal acquisition and transmission; and a Raspberry Pi 5 which executes the Python code. We attached a screen to the system to visualize the Power Spectral Density (PSD) for each execution cycle showing that our transmission, modulated on the predicted frequency slot, does not interfere with other signals in the same wideband.

The CNN-based spectrum occupancy prediction model was converted into a Tensorflow Lite version. This approach produces lighter weights for the model, which are more suitable for the Raspberry Pi. After receiving the index from the spectrum occupancy prediction stage, the filtered complex-valued vector is used for the spectrogram computation using only low-level functions from the *numpy* library in Python [47]. The FFT was computed using a Hanning window (256 size) and the resulting column vectors were arranged to create the spectrogram matrix. This array was then normalized to the 0-to-1 range and mapped to the viridis colormap. After this conversion, the input values are scaled by 255 and converted to the uint8 format. Finally, the RGBA values, created by viridis, were transformed to BGR to comply with the input images from the spectrogram dataset used to train the YOLOv8n classifier. Smoothing of the spectrogram using the BGF was applied just before running the detection operation using the YOLOv8n model. The YOLOv8n model was trained using the same GPU as the CNN-based model in the prediction stage. The stream mode for YOLOv8 was used to pass the spectrogram images and the confidence level was set to 0.6 (i.e., detections with lower confidence are discarded) after heuristic adjustment.

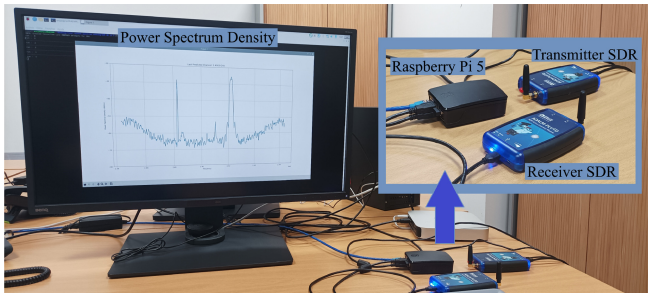

FIGURE 9. Setup of the Raspberry Pi implementation in our laboratory.

The transmission module is only enabled when the RF frame detector does not detect any signal in the selected frequency slot. When this occurs, the spectrum-sharing module connects to a second ADALM-PLUTO SDR unit for data transmission. At the software level, the two SDR modules are easily identified by their distinct IP addresses, allowing a fully customized control of each of them. Our solution, as stated in the introduction, employs the half-duplex operation (HDCR), meaning that the transmission is performed after sensing (spectrum prediction and RF frame detection) using the LBT protocol. However, though it might directly point towards a serialized execution of the different software modules, we have improved the cycle time using a multithreading coding style described in the next section.

B. IMPLEMENTATION MEASUREMENTS

1) THE NUMBER OF FREQUENCY SLOTS

Since the maximum sampling rate of the SDR in this work is 60 MHz, dividing the bandwidth into 12, 30, and 60 frequency slots is very convenient if we want to transmit information using Wi-Fi (2.4 GHz 802.11), BLE 2, and BLE 1 standards:

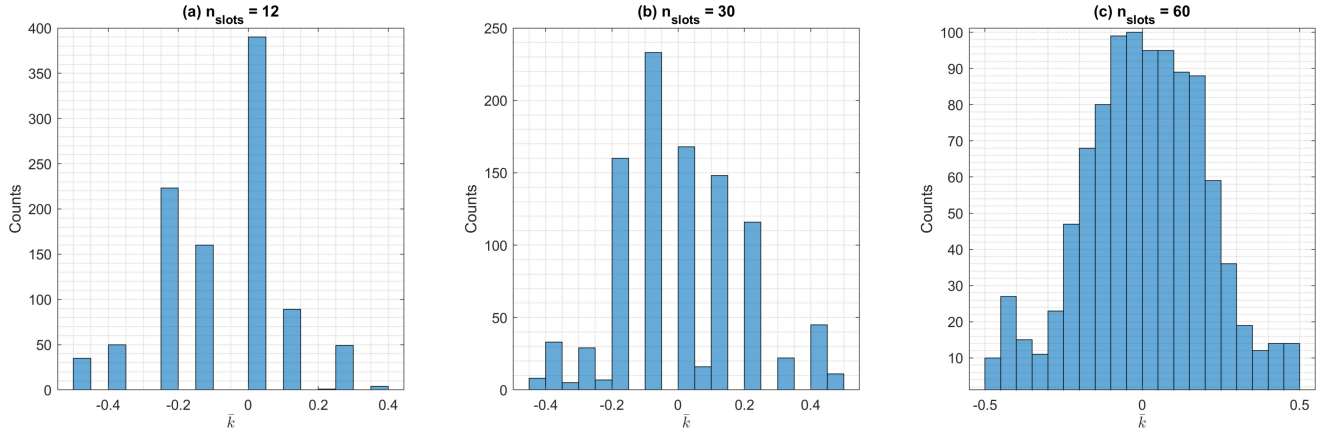

FIGURE 10. Distribution of $\bar{k}$ values for 1000 executions using three n_{slots} values.

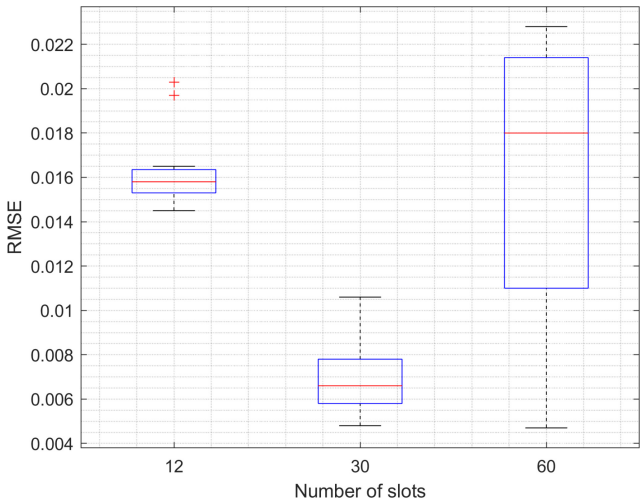

FIGURE 11. Boxplots comparing the RMSE values obtained for the three values of n_{slots}.

- WLAN/Wi-Fi: The IEEE 802.11 standard defines channels whose center frequencies are spaced in 5 MHz steps over the 2.4 GHz ISM band [48], [49]. Since our SDR covers a 60MHz bandwidth, dividing it according to the specified spacing in the IEEE 802.11 results in $n_{slots} = 60/5 = 12$.
- BLE: Bluetooth Low Energy (BLE) defines 40 channels (3 for advertising / 37) with 2 MHz separation [50], [51]. Thus, in this case, we would obtain, $n_{slots} = 60/2 = 30$.
- BT: Classic Bluetooth defines 79 channels with 1 MHz spacing, leading to $n_{slots} = 60/1 = 60$.

Using these three n_{slots} configurations, we evaluated the variability of predicted indexes using 1000 previously recorded I/Q sequences. The results are illustrated in Fig. 10. In general, the system predicted indexes in the center of the band more frequently. To compare the results for the different n_{slots} values, let us express predictions in terms of the dimensionless number $\bar{k}$, which defines how near/far (relative to the total number of slots) from the center of the band the prediction is:

$$\bar{k} = \frac{P_{slot} - C(n_{slots})}{n_{slots}} \tag{1}$$

where P_{slot} is the index of the predicted slot $\in [0, n_{slots} - 1]$, and $C(n_{slots})$ the value[2] that corresponds to the center of the band $C(n_{slots}) = 1/2(n_{slots} - 1)$. Fig. 10 shows the distribution of $\bar{k}$ for 1000 executions. For all the cases, the highest count corresponds to $\bar{k}$ values near 0, meaning closer to the center frequency in the band. When $n_{slots} = 12$, the highest number of predictions occur for index 7, i.e., $\bar{k} = 0.125$. For $n_{slots} = 30$, the more often predicted index is 11, i.e., $\bar{k} = -0.116$. Finally, for $n_{slots} = 60$, the most predicted index was 24, i.e., $\bar{k} = -0.091$. Since the counts were performed over the same I/Q sequences, Fig. 10 allows us to assess the system's performance for index predictions. A variety of indexes are predicted depending on the number of slots. However, simply by comparing the count distribution we can say that the predictions accommodate the real-data spectrum conditions no matter what value for n_{slots} was selected.

2) PREDICTION ACCURACY

The qualitative examination of the results in the previous section only allows us to infer that our solution works properly in general. Nevertheless, we have evaluated the root mean square error (RMSE) between predicted and ground truth indexes to quantify the predictions' accuracy. Fig. 11 compares the resulting RMSE for the three values of n_{slots}. It can be noticed that the RMSE is lower for $n_{slots} = 30$ meaning that predictions using 30 frequency slots were more accurate. When $n_{slots} = 60$, the average RMSE is similar but exhibits a wider standard deviation due to the higher number of possible indexes to be predicted.

3) TIME EXECUTION MEASUREMENTS

Inspired by our previous research [3] and to optimize speed, we implemented a multithreading ($n_{th} = 4$) software

[2]It can be either an integer or have a remainder of 0.5, depending on whether the number of slots is odd or even.

TABLE 3. Execution times (in milliseconds) as a function of n_{slots}.

Component	Max time			Min time			Average time		
	n_{slots}=12	n_{slots}=30	n_{slots}=60	n_{slots}=12	n_{slots}=30	n_{slots}=60	n_{slots}=12	n_{slots}=30	n_{slots}=60
Spectrum prediction	31.194	39.613	54.948	27.228	34.415	46.101	27.799	35.004	46.833
Spectrogram	2.810	3.360	3.474	2.242	2.265	2.264	2.425	2.475	2.623
YOLOv8	0.153	0.148	0.174	0.095	0.095	0.096	0.116	0.118	0.121
Total	31.507	39.929	55.277	27.518	34.702	46.399	**28.101**	35.305	47.131

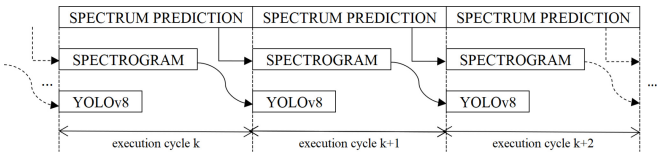

FIGURE 12. Execution cycles using multithreading.

solution with a specific thread dedicated to each of the following components: (th1) spectrum occupancy prediction, (th2) spectrogram calculation, (th3) YOLOv8n inference, (th4) main and sync (execution time negligible). Fig. 12 illustrates (not to scale) the signal processing threads (sync is not shown) corresponding to each component and their behavior over execution cycles. Multithreading maximizes the throughput of the entire system by avoiding unnecessary waiting periods at the expense of a certain lag, which is quickly recovered as the threads synchronize and stabilize over subsequent execution cycles. The resulting cycle time is determined by the longest thread's duration (th1 in our case), rather than by the accumulation of individual durations. Table 3 reports the execution times for each thread in our framework. We compare the times considering the three values of n_{slots}. It is noticeable that as the number of slots increases, the execution time for th1 also increases simply because there are more complex models to run and more predictions to be made. The *Total* row in Table 3 represents the time after the three processes are completed and the system is ready for a new cycle. The smallest mean execution time is 28.101 milliseconds and occurs for $n_{slots} = 12$.

Fig. 13 illustrates the execution time for each thread as a function of the number of frequency slots (n_{slots}) to be predicted. For $n_{slots} = 60$, thread 1 takes the longest to produce a prediction. This makes sense since in this case, the network is more complex and the inference needs to be applied to each slot separately. Fig. 13(b) shows the execution times for thread 2 (spectrogram computation). For this component, the average time is quite similar in all the cases, showing just a slight dependence on the number of slots due to the predicted index's frequency shifting. Finally, for thread 3 (YOLOv8n inference), in Fig. 13(c), the execution times are also very slightly dependent on the number of slots. Table 3 reports the overall cycle time for the system showing how it is mostly conditioned by thread 1 as shown by the Pearson correlation coefficient $\rho = 0.9972$.

4) SENSING RANGE AND POWER CONSUMPTION

To clarify our system's practical scope and limitations, we detail the sensing range and power consumption used for prediction-based SS. The sensing range will depend on multiple factors such as receiver gain, sample rate, and external environment interferences (reflection, refraction, shadowing, etc.). Since the SDR used in this proposal captures over-the-air signals and we assess the performance in an uncontrolled environment, the sensing range will mainly depend on the receiver antenna gain, which was configured manually at 20 dB. The sensing bandwidth for our system covers a range of 60 MHz, close to the highest possible sample rate provided by the ADALM-PLUTO [15].

Since our target is focused on IoT applications, it is important to include insights on energy efficiency, especially given the deployment on a Raspberry Pi and the potential use in battery-powered environments. An estimated power consumption measurement was performed using the *psutil* (process and system utilities) library in Python to retrieve information on running processes and system utilization (CPU, memory, disks, network, etc.). We have employed the *cpu_percent* function to sample the percentage of CPU utilization every second ($CPU_{\%}$) to compute the power consumption (p) using (2).

$$p = p_{idle} + \frac{CPU_{\%} * (p_{full} - p_{idle})}{100} \tag{2}$$

where p_{idle} is the idle power and p_{full} is the full power consumption of the Raspberry Pi 5. Both values were extracted from available data in [52] and [53], leading us to assume that $p_{idle} = 3W$ and $p_{full} = 10W$.

Fig. 14 shows the average power consumption using the three frequency slot configurations. It is interesting to notice that the power consumption is similar in all the cases. The first 10 seconds register the idle power consumption before executing the script. At second 11, the power consumption increases abruptly since the CPU is used by our Python script. Some spikes are seen momentarily since Python opens a new window to show the PSD and the predictions on screen. Finally, approximately at the second 30, the system stabilizes at 4.8 W ($\approx$ 25.7% CPU), which is the average power consumption required by our IoT solution.

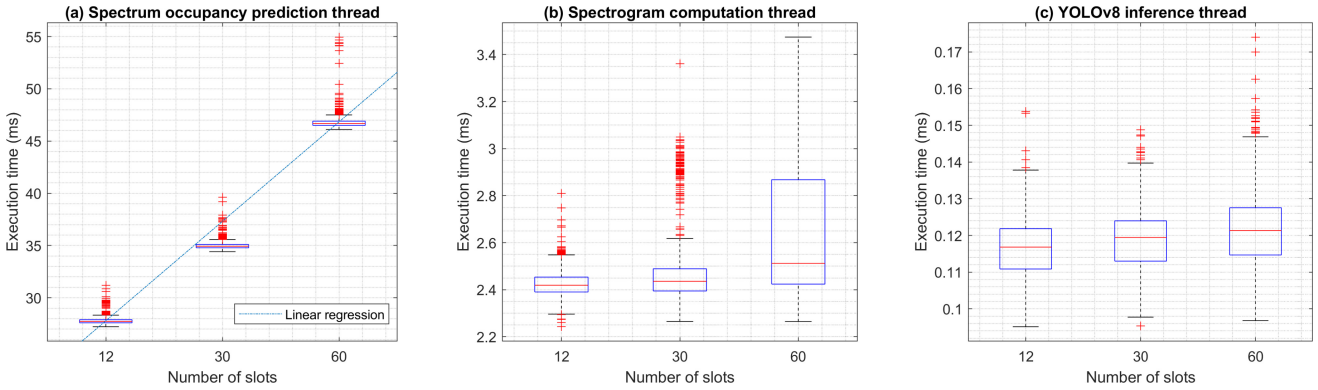

FIGURE 13. Execution times of each thread for the different number of frequency slots.

We understand that an order of several watts is still unrealistic for battery-powered IoT nodes that must operate for extended periods. However, there is no standard power budget for typical IoT scenarios. For example, [54] investigated, on a single and multiple homogeneous edge nodes, the variability of power consumption in various scenarios. In this case, the nodes were multiple Raspberry Pi 3B, as these devices are highly studied in the literature for the prototyping and deployment of IoT and edge systems [55], [56]. The authors' highest power consumption when the edge node is performing an intensive task was $\approx$ 2.70 W, which is similar to our system's power consumption. Currently, the implementation of Raspberry Pi for IoT applications has not covered spectrum sensing. We believe that our power budget can easily accommodate real IoT scenarios by making some improvements concerning power sources and/or internal processor configurations. We suggest the following strategies to reduce power consumption:

- Since transmission is the most power-consuming task, a promising technique called compressive sensing provides an efficient solution by reducing the amount of data collected by sensors [57] (in this case, SDRs). This strategy reduces the amount of data to be transmitted, and consequently the power consumption.
- Another approach to minimize energy consumption involves minimizing the peripherals connected to the Raspberry Pi (for example, the HDMI controller, USB mouse, etc.), installing lighter operating systems (such as Raspbian Lite), and down-clocking its CPU, SDRAM, and GPU units. By modifying these parameters, a saving of 20.1% is obtained compared to the power consumption in idle state [58].
- Although IoT is generally based on battery-based energy use, hybrid energy solutions that combine traditional batteries with renewable sources such as solar energy have the potential to extend the useful life of IoT devices. Hybrid solutions also encourage global sustainability goals by reducing dependence on conventional power sources [59].

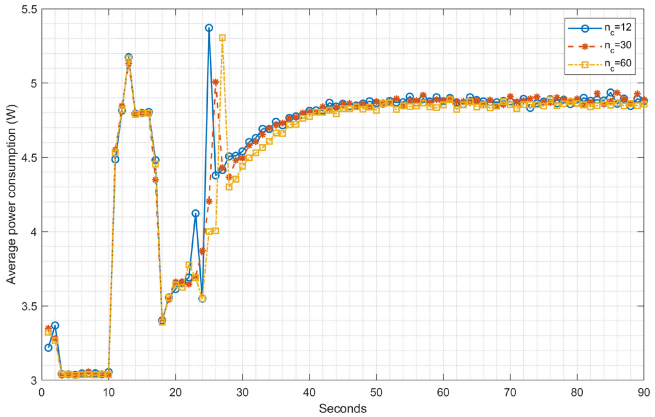

FIGURE 14. Average power consumption using the three frequency slots.

- Finally, the application of simulation tools could validate the proposed strategies and assess their impact in various scenarios of the IoT network, ensuring that the framework remains robust and adaptable in terms of energy efficiency [59].

V. DISCUSSION

Our proposal can be used in multiple frequency bands and environments. Of course, accounting for the ADALM-PLUTO SDR frequency coverage [325 MHz, 3.8 GHz] (by default) [15]. We opted for this SDR because of its affordable price and simple interface communication with Python. Other SDRs could be explored as future directions, such as HackRF, BladeRF, or Universal Software Radio Peripheral (USRP), which might inherently increase the frequency range. In terms of testing environments, by using an SBC such as the Raspberry Pi, we would not depend on voluminous power sources, meaning that our system could potentially be implemented in different laboratories, indoor facilities, or even outdoor facilities using external power banks.

Table 4 compares our proposal with recent spectrum prediction approaches. It is important to clarify beforehand that, obviously, each work utilizes the dataset that best suits its specific requirements. For example, [31] proposed

TABLE 4. Comparison with prior art.

Article	Dataset	DL approach	Complexity			
			Inference time	Processor		
				Training	Testing	IoT suitable?
[7]	RWTH Aachen University spectrum data	Bi-ConvLSTM, Attention, Dense	N/A	N/A	N/A	N/A
[27]	Radio spectrum data (Hangzhou, China)	LSTM, Attention	N/A	GTX Titan X	GTX Titan X	No
[28]	Spectrum occupancy states (NY and Vienna)	CNN, GRU	N/A	N/A	N/A	N/A
[29]	Real-world spectrum data (Istanbul, Turkiye)	LSTM	N/A	GTX 1050	GTX 1050	No
[30]	Sensor measurements	CNN, ResNet	N/A	N/A	N/A	N/A
[31]	Synthetic drone frequency hopping signals	CNN, LSTM	24.7 ms	RTX 2080	RTX 2080	No
[33]	Real measurements (Doha, Qatar)	GRU	N/A	GTX 1650	GTX 1650	No
[34]	Real measurements (Salt Lake City, Utah)	LSTM	N/A	N/A	N/A	N/A
[35]	Aachen University spectrum data	LSTM	8.9 ms	N/A	N/A	N/A
[36]	Synthetic and real occupancy data	CNN, LSTM, CNN-LSTM	31.25 ms	GTX 1650	GTX 1650	No
[37]	2FSK, QPSK	LSTM, Bi-LSTM, ResNet, CNN	1.1 s	N/A	N/A	N/A
[38]	450 MHz-451 MHz band spectrum data	LSTM, Attention	N/A	N/A	N/A	N/A
This work	**Real spectrum data (Seville, Spain)**	**CNN, Dense**	**28.1 ms**	**RTX 3060**	**ARM Cortex A76**	**Yes**

a system targeting drone applications, [30] combined sensor measurements from a wireless sensor network to address spectrum prediction, and [37] evaluated directly digital modulations (2FSK and QPSK). The rest of the papers in this comparison simulated or captured spectrum measurements using an SDR or similar resources. Our system is included in this group.

In general, LSTM models are mostly used for time series-based systems as seen in [27], [29], [31], [34], [35], [36] and [37]. However, CNN-based predictors also have applications similar to our proposal as shown in [28], [30], [31], [36], [37]. More intricate DL approaches such as CNN-LSTM or Bi-ConvLSTM were recently proposed in [36] and [7], respectively. Although these works produce accurate results, they increase the complexity and inference time as seen in Table 4. It is important to emphasize that many proposals do not report inference time for their predictors, thus making the comparison necessarily incomplete. Only [31], [35], [36] and [37] include this information. Our shortest inference time of 28.1 ms corresponds to the system using $n_{slots} = 12$, meaning that we divide the 60 MHz sensing band into twelve 5 MHz slots. This inference time is a reasonable value if we compare it against the 8.9 ms from [35] and the 1.1 s from [37].

Multiple authors have proposed spectrum prediction systems; however, these either provide only simulation results or rely on powerful GPUs, which are inherently unsuitable for IoT applications [27], [29], [31], [33], [36]. Upon reviewing the literature for CR implementations suitable for IoT, we found that according to the comprehensive review of Raspberry Pi applications in [60], only the work by [61] has implemented a CR testbed using a Raspberry Pi 3 with a USRP. However, this implementation focuses on spectrum sensing based on energy detection, rather than a DL-based methodology. To the best of our knowledge, our proposal is the first to implement spectrum prediction for CR applications on a Raspberry Pi 5 (ARM Cortex-A76 processor). Similarly, the review in [62] summarizes recent DL-based spectrum prediction proposals, highlighting the lack of any published CR implementation for spectrum prediction, and, in our opinion, emphasizing the novelty of our contribution.

VI. CONCLUSION

To the best of our knowledge, this is the first work to focus on the implementation of a spectrum prediction proposal devoted to CR applications using an SBC such as the Raspberry Pi 5. This system is an extended version of the DL-based architecture proposed in [3], including two new modules; spectrum prediction occupancy and spectrum sharing. We have proposed a DL-based framework for prediction-based spectrum sensing composed of three stages. First, a spectrum occupancy predictor based on 1D-CNNs predicts the best index for a possible SU transmission. After this, a YOLOv8-based RF frame detector filters and creates a spectrogram from SDR I/Q samples to detect Wi-Fi or BT

signals. The spectrum-sharing stage is only enabled when the RF frame detector does not recognize RF frames allowing the transmission of a test signal in the corresponding predicted frequency slot. The spectrum occupancy prediction network was trained with real-world data representing the spectrum occupancy over time with a 60 MHz bandwidth. For the RF frame detector, a synthetic spectrogram dataset was used for transfer learning on a YOLOv8 nano model. The shortest average execution time for our proposal was 28.101 ms using $n_{slots} = 12$ (twelve equally spaced frequency slots across the 60 MHz bandwidth). These results open the doors to future hardware implementations of DL-based modules for CR applications based on transceivers or IoT devices.

REFERENCES

[1] X. Cao et al., "AI-empowered multiple access for 6G: A survey of spectrum sensing, protocol designs, and optimizations," *Proc. IEEE*, vol. 112, no. 9, pp. 1264–1302, Sep. 2024.

[2] H. N. Nguyen, M. Vomvas, T. Vo-Huu, and G. Noubir, "Wideband, real-time spectro-temporal RF identification," in *Proc. 19th ACM Int. Symp. Mobil. Manag. Wireless Access*, 2021, pp. 77–86.

[3] A. Rojas, G. Liñán-Cembrano, G. J. Dolecek, and J. de la Rosa, "Deep learning-based architecture for RF frame detection for CR applications using spectrograms," in *Proc. IEEE 67th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, 2024, pp. 122–125.

[4] J. Mitola, "The software radio architecture," *IEEE Commun. Mag.*, vol. 33, no. 5, pp. 26–38, May 1995.

[5] J. Mitola, "Cognitive radio architecture evolution," *Proc. IEEE*, vol. 97, no. 4, pp. 626–641, Apr. 2009.

[6] A. Nasser, H. Al Haj Hassan, J. Abou Chaaya, A. Mansour, and K.-C. Yao, "Spectrum sensing for cognitive radio: Recent advances and future challenge," *Sensors*, vol. 21, no. 7, p. 2408, 2021.

[7] L. Wang, J. Hu, R. Jiang, and Z. Chen, "A deep long-term joint temporal–spectral network for spectrum prediction," *Sensors*, vol. 24, no. 5, p. 1498, 2024.

[8] A. Rojas, G. J. Dolecek, and J. M. de la Rosa, "Addressing preprocessing for spectrum sensing using image processing," *Digit. Signal Process.*, vol. 156, Jan. 2025, Art. no. 104800.

[9] H. Eltom, S. Kandeepan, R. J. Evans, Y. C. Liang, and B. Ristic, "Statistical spectrum occupancy prediction for dynamic spectrum access: A classification," *EURASIP J. Wireless Commun. Netw.*, vol. 29, pp. 1–17, Feb. 2018.

[10] X. Xing, T. Jing, W. Cheng, Y. Huo, and X. Cheng, "Spectrum prediction in cognitive radio networks," *IEEE Wireless Commun.*, vol. 20, no. 2, pp. 90–96, Apr. 2013.

[11] T. O'Shea, T. Roy, and T. C. Clancy, "Learning robust general radio signal detection using computer vision methods," in *Proc. 51st Asilomar Conf. Signals, Syst., Comput.*, 2017, pp. 829–832.

[12] G. Jocher, A. Chaurasia, and J. Qiu. "Ultralytics YOLOv8." 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[14] B. Bhavya Sree, V. Yashwanth Bharadwaj, and N. Neelima, "An inter-comparative survey on state-of-the-art detectors—R-CNN, YOLO, and SSD," in *Intelligent Manufacturing and Energy Sustainability*. Singapore: Springer, 2021, pp. 475–483.

[15] F. Nešković, N. Basta, and M. P. Ivaniš, "ADALM-Pluto SDR as a tool for S-parameter estimation: A comparative study of various system identification procedures," in *Proc. 10th Int. Conf. Electr., Electron. Comput. Eng. (IcETRAN)*, 2023, pp. 1–5.

[16] K. N. R. S. V. Prasad, K. B. D'souza, and V. K. Bhargava, "A downscaled faster-RCNN framework for signal detection and time-frequency localization in wideband RF systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4847–4862, Jul. 2020.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–14.

[18] N. Soltani, V. Chaudhary, D. Roy, and K. Chowdhury, "Finding waldo in the CBRS band: Signal detection and localization in the 3.5 GHz spectrum," in *Proc. IEEE Global Commun. Conf.*, 2022, pp. 4570–4575.

[19] S. Basak, S. Rajendran, S. Pollin, and B. Scheers, "Combined RF-based drone detection and classification," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 1, pp. 111–120, Mar. 2022.

[20] P. M. Mutescu, A. Lavric, A. I. Petrariu, and V. Popa, "A hybrid deep learning spectrum sensing architecture for IoT technologies classification," in *Proc. 17th Int. Conf. Eng. Modern Electr. Syst. (EMES)*, 2023, pp. 1–4.

[21] R. Zhao, Y. Ruan, and Y. Li, "Cooperative time-frequency localization for wideband spectrum sensing with a lightweight detector," *IEEE Wireless Commun. Lett.*, vol. 27, no. 7, pp. 1844–1848, Jul. 2023.

[22] R. Zhao, Y. Ruan, Y. Li, T. Li, and R. Zhang, "CCD-GAN for domain adaptation in time-frequency localization-based wideband spectrum sensing," *IEEE Wireless Commun. Lett.*, vol. 27, no. 9, pp. 2521–2525, Sep. 2023.

[23] S. Kayraklik, I. Yildirim, E. Basar, I. Hokelek, and A. Gorcin, "Practical implementation of RIS-aided spectrum sensing: A deep-learning-based solution," *IEEE Syst. J.*, vol. 18, no. 2, pp. 1481–1488, Jun. 2024.

[24] A. Olesiński and Z. Piotrowski, "A radio frequency region-of-interest convolutional neural network for wideband spectrum sensing," *Sensors*, vol. 23, no. 14, p. 6480, 2023.

[25] A. Vagollari, M. Hirschbeck, and W. Gerstacker, "An end-to-end deep learning framework for wideband signal recognition," *IEEE Access*, vol. 11, pp. 52899–52922, 2023.

[26] A. Mittal et al., "Sub-6-GHz energy-detection-based fast on-chip analog spectrum sensing with learning-driven signal classification," *IEEE Internet Things J.*, vol. 11, no. 14, pp. 25033–25046, Jul. 2024.

[27] K. Li, Z. Liu, S. He, and J. Chen, "TF2AN: A temporal-frequency fusion attention network for spectrum energy level prediction," in *Proc. 16th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, 2019, pp. 1–9.

[28] L. Yu et al., "Spectrum availability prediction for cognitive radio communications: A DCG approach," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 2, pp. 476–485, Jun. 2020.

[29] M. A. Aygül et al., "Spectrum occupancy prediction exploiting time and frequency correlations through 2D-LSTM," in *Proc. IEEE 91st Veh. Technol. Conf.*, 2020, pp. 1–5.

[30] X. Ren, H. Mosavat-Jahromi, L. Cai, and D. Kidston, "Spatio-temporal spectrum load prediction using convolutional neural network and ResNet," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 502–513, Jun. 2022.

[31] S. Basak, S. Rajendran, S. Pollin, and B. Scheers, "Spectrum prediction for protocol-aware RF jamming," *IEEE Trans. Cogn. Commun. Netw.*, vol. 10, no. 2, pp. 363–373, Apr. 2024.

[32] H. Mosavat-Jahromi, Y. Li, L. Cai, and J. Pan, "Prediction and modeling of spectrum occupancy for dynamic spectrum access systems," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 3, pp. 715–728, Sep. 2021.

[33] A. Tusha, B. Kaplan, H. A. Cırpan, K. Qaraqe, and H. Arslan, "Intelligent spectrum occupancy prediction for realistic measurements: GRU based approach," in *Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, 2022, pp. 179–184.

[34] A. Ghosh, S. Kasera, and J. Van Der Merwe, "Spectrum usage analysis and prediction using long short-term memory networks," in *Proc. 24th Int. Conf. Distrib. Comput. Netw.*, 2023, pp. 270–279.

[35] R. Nandakumar, V. Ponnusamy, and A. K. Mishra, "LSTM Based spectrum prediction for real-time spectrum access for IoT applications," *Intell. Autom. Soft Comput.*, vol. 35, no. 3, p. 2805, 2023.

[36] P. I. Enwere, E. Cervantes-Requena, L. A. Camunas-Mesa, and J. M. de la Rosa, "Using ANNs to predict the evolution of spectrum occupancy in cognitive-radio systems," *Integration*, vol. 93, Nov. 2023, Art. no. 102070.

[37] L. Xu et al., "Spectrum prediction for mobile Internet of Things based on a DB-LSTM algorithm," *IEEE Trans. Veh. Technol.*, vol. 73, no. 10, pp. 15395–15406, Oct. 2024.

[38] S. Chen, B. Xia, Y. Tang, and J. Zhu, "Multi-band spectrum prediction using combined model for power wireless private networks," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC Workshops)*, 2024, pp. 448–452.

[39] J. Kim, S. Oh, H. Kim, and W. Choi, "Tutorial on time series prediction using 1D-CNN and BiLSTM: A case example of peak electricity demand and system marginal price prediction," *Eng. Appl. Artif. Intell.*, vol. 126, Nov. 2023, Art. no. 106817.
[40] J. Wicht, U. Wetzker, and V. Jain, "Spectrogram data set for deep-learning-based RF frame detection," *Data*, vol. 7, no. 12, p. 168, 2022.
[41] S. Garnier, N. Ross, B. Rudis, M. Sciaini, A. P. Camargo, and C. Scherer. "viridis(Lite)—Colorblind-friendly color maps for R, Viridis package version 0.6.5." 2024. [Online]. Available: https://sjmgarnier.github.io/viridis/
[42] A. Rojas, G. J. Dolecek, J. De La Rosa, and G. Liñán-Cembrano, "Increasing the accuracy of spectrogram-based spectrum sensing trained by a deep learning network using a Resnet-18 model," in *Proc. 39th Conf. Design Circuits Integr. Syst. (DCIS)*, 2024, pp. 1–4.
[43] S. Bhattarai, J.-M. J. Park, B. Gao, K. Bian, and W. Lehr, "An overview of dynamic spectrum sharing: Ongoing initiatives, challenges, and a roadmap for future research," *IEEE Trans. Cogn. Commun. Netw.*, vol. 2, no. 2, pp. 110–128, Jun. 2016.
[44] H. Nasiri, S. Tusha, M. I. Rochman, and M. Ghosh, "Data driven environment classification using wireless signals," in *Proc. 30th Annu. Int. Conf. Mobile Comput. Netw.*, 2024, pp. 1906–1913.
[45] R. Biswas, J. Wu, X. Du, and Y. Yang, "Mitigation of the spectrum sensing data falsifying attack in cognitive radio networks," *Cyber-Phys. Syst.*, vol. 7, no. 3, pp. 159–178, 2021.
[46] L. Miao, X. Di, Z.-M. Huo, and Z.-X. Sun, "Research on spectrum sensing data falsification attack detection algorithm in cognitive Internet of Things," *Telecommun. Syst.*, vol. 80, pp. 227–238, Jun. 2022.
[47] C. R. Harris et al., "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sep. 2020.
[48] A. Zankiewicz, "Susceptibility of IEEE 802.11n networks to adjacent-channel interference in the 2.4 GHz ISM band," *Przeglad Elektrotechniczny*, vol. 88, no. 9b, pp. 287–288, 2012.
[49] C. J. Bernardos, I. Soto, and A. Banchs, *IEEE Standard for Information technology—Local and metropolitan area networks—Specific requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Fast Basic Service Set (BSS) Transition*, IEEE Standard 802.11, 2008.
[50] R. Natarajan, P. Zand, and M. Nabi, "Analysis of coexistence between IEEE 802.15.4, BLE and IEEE 802.11 in the 2.4 GHz ISM band," in *Proc. 42nd Annu. Conf. IEEE Ind. Electron. Soc.*, 2016, pp. 6025–6032.
[51] P. Di Marco, R. Chirikov, P. Amin, and F. Militano, "Coverage analysis of Bluetooth low energy and IEEE 802.11ah for office scenario," in *Proc. IEEE 26th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, 2015, pp. 2283–2287.
[52] P. Steadman, P. Jenkins, R. S. Rathore, and C. Hewage, "Challenges in Implementing Artificial Intelligence on the Raspberry Pi 4, 5 and 5 with AI HAT," in *Proc. Int. Conf. Comput., Commun., Cybersecur. AI*, 2024, pp. 147–157.
[53] S. Bast, L. Begic Fazlic, S. Naumann, and G. Dartmann, "LLMs on the edge: Quality, latency, and energy efficiency," in *Proc. INFORMATIK*, 2024, pp. 1183–1192.
[54] S. Tofaily, I. Raïs, and O. Anshus, "Quantifying the variability of power and energy consumption for IoT edge nodes," in *Proc. 19th Int. Conf. Distrib. Comput. Smart Syst. Internet Things (DCOSS-IoT)*, 2023, pp. 577–584.
[55] M. Saari, A. M. Bin Baharudin, and S. Hyrynsalmi, "Survey of prototyping solutions utilizing Raspberry Pi," in *Proc. 40th Int. Convent. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, 2017, pp. 991–994.
[56] C. N. Cabaccan, F. Reidj, and G. Cruz, "Power characterization of Raspberry Pi agricultural sensor nodes using Arduino based voltmeter," in *Proc. 3rd Int. Conf. Comput. Commun. Syst. (ICCCS)*, 2018, pp. 349–352.
[57] M. Capra, R. Peloso, G. Masera, M. Ruo Roch, and M. Martina, "Edge computing: A survey on the hardware requirements in the Internet of Things world," *Future Internet*, vol. 11, no. 4, p. 100, 2019.
[58] F. Astudillo-Salinas, D. Barrera-Salamea, A. Vázquez-Rodas, and L. Solano-Quinde, "Minimizing the power consumption in Raspberry Pi to use as a remote WSN gateway," in *Proc. 8th IEEE Latin-Amer. Conf. Commun. (LATINCOM)*, 2016, pp. 1–5.
[59] Z. Almudayni, B. Soh, H. Samra, and A. Li, "Energy inefficiency in IoT networks: Causes, impact, and a strategic framework for sustainable optimisation," *Electronics*, vol. 14, no. 1, p. 159, 2025.
[60] S. E. Mathe, H. K. Kondaveeti, S. Vappangi, S. D. Vanambathina, and N. K. Kumaravelu, "A comprehensive review on applications of Raspberry Pi," *Comput. Sci. Rev.*, vol. 52, May 2024, Art. no. 100636.
[61] H. J. Park, G.-M. Lee, S.-H. Shin, B.-H. Roh, and J. M. Oh, "Implementation of multi-hop cognitive radio testbed using Raspberry Pi and USRP," *Int. J. Interdiscipl. Telecommun. Netw.*, vol. 9, no. 4, pp. 37–48, 2017.
[62] L. Wang, J. Hu, D. Jiang, C. Zhang, R. Jiang, and Z. Chen, "Deep learning models for spectrum prediction: A review," *IEEE Sensors J.*, vol. 24, no. 18, pp. 28553–28575, Sep. 2024,

ANDRES ROJAS (Graduate Student Member, IEEE) received the bachelor's degree in electronics and telecommunications engineering from the University of Cuenca, Ecuador, in 2018, and the M.Sc. degree in electronics from the National Institute of Astrophysics, Optics, and Electronics, Puebla, Mexico, in 2021. He is currently pursuing the Ph.D. degree. He is a Research Intern with the Institute of Microelectronics of Seville, Spain. He is the author of one book chapter for *Encyclopedia of Information, Science, and Technology, Sixth Edition*, four journal papers, and seven conference papers. His research interests include digital signal processing, image processing, artificial intelligence, wireless communications, and cognitive radio.

GAWEN FOLLET received the bachelor's degree in microelectronics and automation engineering from the Polytechnic Montpellier in 2024. In 2023, he did a Summer Internship with the Institute of Microelectronics of Seville, where he worked on the implementation of artificial neural networks in cognitive radio systems.

GORDANA JOVANOVIC DOLECEK (Life Senior Member, IEEE) received the B.Sc. degree from the University of Sarajevo, Bosnia and Herzegovina, the M.Sc. degree from the University of Beograd, Serbia, and the Ph.D. degree from the University of Sarajevo. In 1995, she joined INAOE, Department for Electronics, Puebla, Mexico, where she works as a Full Professor. She was a Visiting Researcher with UCSB, Santa Barbara, USA, from 2001 to 2002 and in 2006, with SDSU San Diego, USA, from 2008 to 2009, and also with UCLA, Los Angeles, USA, from 2015 to 2016. She is the author/coauthor of more than 80 journals and 400 conference papers. She is the author of the book: *Random Signals and Processes Primer with MATLAB* (Springer, NY, USA: 2013), and editor of books: *Advances in Multirate Systems* (Springer, NY, USA, 2017), and *Multirate Systems: Design and Applications* (IGP, Hershey, USA, 2001). Her research interests include digital signal processing (DSP), digital communications, education methods for DSP and digital communications, machine learning, and deep learning for DSP and digital communications. In 2024, she received the 2024 IEEE Circuits and Systems Society John Choma Education Award; the Best Associate Editor IEEE TCAS II Award, 2023; and in 2023, the Best Associate Editor IEEE TCAS I Award in 2022, and from 2020 to 2021. In 2022, she was the recipient of the IEEE Puebla Award "Mujer en Ciencia." In 2012, she received the Science and Technology Puebla State Award for her research work in electronics. She was an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEF, and *IEEE Circuits and Systems Magazine*. She is currently a Senior Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEF and *IET Signal Processing*. She is a Guest Editor for the Special Issue "Advances in Signal Processing and Artificial Intelligence for Wireless Communications," *Applied Science*. She was a Guest Editor for the Special Issue "Advances of Wireless Communications Using Machine Learning and Deep Learning Techniques," for *Applied Science* (MDPI) in 2023, a Guest Editor with N. Cho, for the Special Issue "Advances in Image Processing Using Machine Learning Techniques," for *IET Signal Processing* in 2022, and one of the Guest Editors for Special Issue: "Digital Divide: Closing the Gap," *IEEE Communication Magazine*. She was a Co-Organizer of a Special Session at ISCAS in Melbourne, Australia, and a tutorial at the IEEE LASCAS Conference in Montevideo, Uruguay. She was a committee member for more than 80 international conferences. She is a member of the CAS DSP Committee and a Review Committee DSP Member for IEEE ISCAS Conferences. She is a TPC Member of the Flagship Latin America CAS Conference LASCAS. She is a member of the Mexican Academy of Sciences, SNI of Mexico.

JOSÉ M. DE LA ROSA (Fellow, IEEE) received the M.S. degree in physics and the Ph.D. degree in microelectronics from the University of Seville, Spain, in 1993 and 2000, respectively.

Since 1993, he has been working with the Institute of Microelectronics of Seville, which is in turn part of the Spanish Microelectronics Center of the Spanish National Research Council. He is also a Full Professor with the University of Seville. His main research interests are in the field of analog and mixed-signal integrated circuits, especially high-performance data converters. In these topics, he has co-authored nearly 300 international publications, including journal and conference papers, book chapters, and six books, the latter being *CMOS Sigma-Delta Converters: Practical Design Guide* (Wiley-IEEE Press, 2nd Edition, 2018). He is on the Stanford University World's Top Scientists List (editions 2019–2024). He is a member of the IEEE Circuits and Systems Society (CASS) and the IEEE Solid-State Circuits Society. He served as a Distinguished Lecturer of IEEE-CASS from 2017 to 2018, and as the Chair of the Spain Chapter of IEEE-CASS from 2016 to 2017. He was at the front of the Editorial Board of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, where he was the Deputy Editor-in-Chief from 2016 to 2019 and the Editor-in-Chief from 2020 to 2021. He also served as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS, where he received the 2012–2013 Best Associate Editor Award. He has been involved in the organizing and technical committees of many IEEE conferences, including ISCAS, MWSCAS, ICECS, LASCAS, VLSI-SoC, DATE, ISICAS, and ESSCIRC. He has been a member of the Executive Committee of the IEEE Spain Section from 2014 to 2015 and from 2016 to 2017, where he served as Membership Development Officer from 2016 to 2017. He is a Member-at-Large of the IEEE-CASS Board of Governors from 2023 to 2025, and serves as the Editor-in-Chief for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I: REGULAR PAPERS from 2024 to 2025.

GUSTAVO LIÑÁN-CEMBRANO received the degree in physics in 1996, and the Ph.D. degree in microelectronics. He is a Tenured Scientist of the Spanish National Research Council, Instituto de Microelectrónica de Sevilla (IMSE-CNM CSIC-Univ. Sevilla). His research career focused at the beginning on the design of high-complexity mixed-signal chips for vision, embedding sensing, and processing at the pixel level. For about ten years he was involved in designing CMOS imagers exhibiting very high dynamic range for industrial applications, including automotive. In recent years, he has focused his research on vision processing applications using artificial intelligence, among other techniques, for ecology applications, where he has published in the top journals, including Science, and a software package that is freely distributed with more than 2000 users all over the world. He has authored more than 130 journal and conference papers. He has received the Best Paper Award of the International Journal of Circuit Theory and Applications two times, and recently the Third Best Live Demo Award at ISCAS'2024.