PU-DetNet: An Efficient Deep Unfolding aided Smart Sensing Framework for Cognitive Radio

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

04-02-2022 / 11-02-2022

CITATION

Shah, Sanket; Patel, Dhaval; Soni, Brijesh; Lopez-Benitez, Miguel; Govindasamy, Siddhartan (2022): PU-DetNet: An Efficient Deep Unfolding aided Smart Sensing Framework for Cognitive Radio. TechRxiv. Preprint. <https://doi.org/10.36227/techrxiv.19119752.v1>

DOI

[10.36227/techrxiv.19119752.v1](https://doi.org/10.36227/techrxiv.19119752.v1)

PU-DetNet: An Efficient Deep Unfolding aided Smart Sensing Framework for Cognitive Radio

Sanket B. Shah, Dhaval K. Patel, *Member IEEE*, Brijesh Soni,

Miguel López-Benítez, *Senior Member, IEEE*,

Siddhartan Govindasamy, *Member, IEEE*

Abstract

Spectrum sensing in cognitive radio (CR) paradigm can be broadly categorized as model/analytical based and data based approaches. However, the former is sensitive to model inaccuracies in evolving network environment, while the latter (machine learning (ML)/deep learning (DL) based approach) suffers from high computational complexity and black box like behaviour. For devices with low computational abilities, such approaches could be rendered less useful. In this context, we propose a deep unfolding architecture namely the Primary User-Detection Network (PU-DetNet) that harvests the strength of both: model and data based approach. In particular, a technique is described that reduces computational complexity. It involves binding the loss function such that each layer of the proposed architecture possesses its own loss function whose aggregate is optimized during training. The proposed PU-DetNet scheme is validated on five different datasets. Compared to the state-of-the-art, experimental results demonstrate that the probability of detection is significantly improved as compared to the long short term memory (LSTM) scheme (between 39% and 56%), convolutional neural network (CNN) scheme (between 45% and 84%) and artificial neural network (ANN) scheme (between 53% and 128%) over empirical, 5G simulated, DeepSig, satellite communications, and radar

Manuscript received on February 03, 2022. This work was supported by the British Council under UKIERI DST Thematic Partnerships 2016-17 (grant ref. DST-198/2017).

S. B. Shah and D. K. Patel are with the School of Engineering and Applied Science, Ahmedabad University, Ahmedabad 380009, India (e-mail: sanket.s1@ahduni.edu.in; dhaval.patel@ahduni.edu.in)

B. Soni is with Boston College, MA 02467, USA (email: sonib@bc.edu)

M. López-Benítez is with Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool L69 3GJ, United Kingdom. He is also with the ARIES Research Centre, Antonio de Nebrija University, 28040 Madrid, Spain (email: m.lopez-benitez@liverpool.ac.uk).

S. Govindasamy is with Boston College, MA 02467, USA (email: govindsi@bc.edu)

dataset. Additionally, computational complexity reduces $\approx$ by 91.69% and 93.15% w.r.t. LSTM and ANN schemes respectively. Moreover, the proposed scheme also shows improvement in throughput by $\approx$ 87% and 130% as compared to the LSTM and ANN based schemes, respectively.

Index Terms

Dynamic spectrum access, Deep Learning, PU-DetNet, Cognitive Radio, Spectrum Sensing, Computational complexity.

I. INTRODUCTION

A. Background and current state-of-the-art

Sensing and detection is envisaged to have numerous applications for fifth generation (5G) and beyond wireless communications, including but not limited to radar detection, integrated sensing and communications, and dynamic spectrum access [1], [2]. With the rapid advancement of wireless communication technologies and the advent of the 5G paradigm, spectrum resources have become highly scarce [3], while the number of connected devices have increased massively.

Cognitive radio (CR) communications [4] have emerged as a potential solution to trade-off between spectrum availability and its demanding growth [5], [6]. CR is defined as “*an intelligent wireless communication system that is aware of its environment and uses the methodology of understanding-by-building to learn from the environment*” [7]. The underlying principle of CR is to allow the unlicensed users to access the temporarily unoccupied licensed bands in an opportunistic and non-interfering manner [8]. This calls for highly reliable and efficient spectrum sensing schemes [7].

State of the art spectrum sensing approaches can be broadly classified into two categories: analytical/model based approach, and data driven approach. The model based approach involves the use of established mathematical techniques and analytical tools for evaluating the spectrum sensing performance. Such works can be found in [9]–[12], and references therein. Although model based approach performs well, it is sensitive to model inaccuracies. Moreover, they may be unsuitable for the evolving network environment [13], especially in 5G and beyond networks. Contrary to model based approach, data driven approach does not incorporate any knowledge

obtained by analysis through mathematical techniques, and relies mostly on the distribution and/or type of data.

With the rapid advancement in the learning based signal processing techniques, data driven approach have gained wide attention from industry and academia in the context of future wireless networks [14]–[17], including spectrum sensing in CR communications [18]. Owing to the excellent learning ability of data-driven approach, many works in literature have leveraged the machine learning (ML) / deep learning (DL) techniques considering spectrum sensing as a binary classification problem. For instance, artificial neural network (ANN) based spectrum sensing was carried out in [19]. A novel ANN-based hybrid sensing scheme was proposed in [20] which used energy values and the Zhang statistics as the training features, and with an improved technique presented in [21]. The sensing of orthogonal frequency division multiplexing (OFDM) signals at a low SNR regime using a Naive-Bayes classifier was proposed in [22]. Recently, ML enabled cooperative spectrum sensing for non-orthogonal multiple access was carried out in [23]. Furthermore, a few works have also applied the DL approach to spectrum sensing. For instance, a convolutional neural network (CNN) based spectrum sensing was proposed in [24]–[26], CNN based cooperative sensing was implemented in [27], while stacked auto-encoder based spectrum sensing of OFDM signal was proposed in [28]. Moreover, long short term memory (LSTM) based spectrum sensing schemes were studied in [29]–[30] while, various other DL architectures for spectrum sensing schemes were carried out in [31]–[34].

B. Motivation

Although the data driven approach aided ML/DL spectrum sensing techniques have shown performance efficiency and unprecedented empirical success, such techniques usually suffer from the requirement of exhaustive training, computationally large training data, explainability of trained ML/DL model, numerous trainable parameters, and high computational complexity. Furthermore, such ML/DL aided frameworks are generally trained, tested and deployed on

an environment powered by a computationally equipped graphics processing unit (GPU). The time taken to produce outputs completely depends upon the specifications and robustness of the hardware. This implies that devices with lesser efficiency and capacity are bound to face difficulties in deploying ML/DL frameworks effectively. Especially for network architectures where it is envisaged that there would be massive devices with low computational abilities, for instance machine-type communications, low cost sensors, and edge devices (also included in the 3rd generation partnership project (3GPP) release-14 and to its subsequent versions [35]), the implementation and usage of a conventional ML/DL framework could be rendered less useful [36].

To overcome the aforementioned shortcomings of both the model and data driven approaches, the concept of deep unfolding was introduced in [37] which aims at simultaneously harvesting the strength of both model and data based approaches. Given a model-based approach, one unfolds the iterations of a derived algorithm into a layer-wise structure analogous to an ML/DL architecture [38] such that each iteration is considered a layer and algorithm is called a network. This approach combines the expressive power of a conventional deep network with the explainability of a mathematical model-based approach. There have been few recent works reported in the literature which have utilised deep unfolding techniques. For instance, the work in [39] attempts to compute the bit error rate for MIMO systems using alternating direction method of multipliers (ADMM) unfolding, while [36] attempts via iterative algorithm induced deep unfolding. Moreover, MIMO detection using Deep unfolding technique was studied in [40]. The work in [41] focused on the deep unfolding for compressive sensing. A comprehensive survey of deep unfolding for wireless communications can be found in [42].

As far as spectrum sensing is concerned, the sensing needs to be accurate and efficient in terms of the detection performance. The aforementioned shortcomings of the data driven approach further add challenges in sensing for devices with low computational abilities. To the best of our knowledge, no work has been reported in the literature which attempts to use deep unfolding

techniques for spectrum sensing. In this context, the key objective of this work is to design a computationally efficient and smart sensing framework that attempts to detect the primary user (PU). A deep unfolding scheme alongside the architectural innovations presented performs spectrum sensing in a computationally inexpensive manner while also yielding promising results, especially in the low SNR regime.

C. Contributions of this paper

The key contributions of our work are four fold and can be summarised as:

- Firstly, we propose a novel architecture namely the primary user detection network (PU-DetNet), where the approach is to iteratively reduce the error between classification and ground truth by unfolding a neural network into layers whose structure differs critically from that of a generic deep neural network.
- Secondly, a technique is described that involves binding the loss function such that each layer of the proposed architecture possesses its own loss function whose aggregate is optimized. This technique leads to a state where after training, values of the loss functions from shallow to deep layers become nearly equal. The implication of this property is that almost all layers of the network can generate outputs that are close to the true labels. Hence, shallow layers have nearly equivalent performance to deeper layers.
- Thirdly, owing to the impact of loss binding, we examine the effect of discarding deeper layers in order to minimise the computational complexity. With experimental proof, the resulting system was observed to be reliable and efficient in PU detection.
- Lastly, the proposed PU-DetNet scheme is experimentally validated with spectrum data from five different datasets: empirical test-bed measurement setup, a 5G simulated dataset, the DeepSig dataset, a satellite communications dataset and a radar dataset. The obtained results show that the probability of detection is improved by a significant amount compared to the LSTM approach (between 39% to 56%), CNN approach (45% to 84%) and the ANN

approach (between 53% and 128%). Moreover, compared to the state-of-the-art LSTM and ANN based sensing schemes, the computational complexity is reduced by 91.69% and 93.15% respectively. Furthermore, the proposed scheme shows improvement in throughput by 87% and 130% more compared to LSTM and ANN based schemes, respectively.

The remainder of this paper is organized as follows: Section II describes the system model and preliminaries of the deep unfolding framework. The architecture of proposed PU-DetNet scheme is proposed in Section III. Section IV provides the description of the considered datasets. Section V comprehensively describes the experimental results. Finally, Section VI concludes this work.

Notations: Boldface uppercase letters represent matrices. Boldface lowercase represent vectors. Lowercase letters represent scalars. Raw outputs (pre-activation) are denoted as $(\cdot)'$. The i^{th} data sample is denoted as $(\cdot)_i$ and the truth label corresponding to the same is denoted as $(\cdot)_i^t$. Variables corresponding to the k^{th} layer of the considered architecture are referred to as $(\cdot)_k$.

II. SYSTEM MODEL AND PRELIMINARIES OF DEEP UNFOLDING

A. Considered System Model

The problem of PU detection can be formulated as a binary classification problem. The secondary user (SU) tries to detect the presence of the PU from the signal it receives, defined as $y^{<t>}$. The binary hypothesis can be written as:

$$\mathcal{H}_0 : y^{<t>} = w^{<t>} \quad (1)$$

$$\mathcal{H}_1 : y^{<t>} = h^{<t>} x^{<t>} + w^{<t>},$$

where $y^{<t>}$ is the received signal by the SU at a time t , $x^{<t>}$ is the transmitted signal and $w^{<t>}$ is additive white Gaussian noise with zero mean and variance σ^2 and $h^{<t>}$ is the flat-fading channel coefficient. $\mathcal{H}_0$, the null hypothesis, describes the scenario when only noise is present during the fixed time sensing event, i.e., indicating the absence of PU, while $\mathcal{H}_1$, the alternate hypothesis, describes the presence of PU.

Fig. 1 demonstrates the generic flow of a deep unfolding framework. Given a model-based approach, one unfolds the iterations of a derived algorithm into a layer-wise structure analogous

Fig. 1: Generic flow of a deep unfolding framework

to an ML/DL architecture [38] such that, each iteration of an algorithm is computed by a different layer in the unfolded architecture. To form such a model, an iterative algorithm is derived and trainable parameters are identified which form the basis of the learning architecture. Instead of optimizing a generic neural network, we untie the trainable parameters of the model across layers to create a more flexible network. The resulting architecture can be trained discriminatively to perform accurate inference within a fixed network size. This approach combines the expressive power of a conventional deep network with the explainability of a mathematical model-based approach.

B. Iterative Approach to PU Detection

The objective is to derive an expression that aims at iteratively reducing the error of estimation¹. Table I lists out the variables used in this paper. Just like the gradient descent equation, we estimate the error and reduce it in a stepwise manner. Such an idea could be expressed as:

$$\mathbf{y}_k = \Pi [\mathbf{y}_{k-1} - \delta_k \mathbf{e}_k], \quad (2)$$

where $\mathbf{y}_{k-1}$ is the estimation in the $(k-1)^{th}$ iteration with respect to the whole signal dataset and $\mathbf{y}_k$ is the estimation in the k^{th} iteration, δ_k is a tunable scaling parameter, $\mathbf{e}_k$ is the error observed in the k^{th} iteration and $\Pi[\cdot]$ is a non linear projection operator. To define the error $\mathbf{e}_k$ in (2), we first define the error for a single data sample:

¹Although PU detection is a classification problem, it is analogous to estimating a function $f: \mathbb{R}^n \rightarrow \{\mathcal{H}_0, \mathcal{H}_1\}$ and assigning a categorical label to the input data [43].

TABLE I: Variables and their description used in this paper

Variable	Description
$\mathbf{X}$	Signal energies dataset
$\mathbf{y}_k$	Estimated outputs of the k^{th} layer
$\mathbf{y}^t$	Ground truth in vector form
$\mathbf{e}_k$	Error value between $\mathbf{y}_k$ and $\mathbf{y}^t$
$\mathbf{x}_i$	Signal energies of the i^{th} data sample
$\boldsymbol{\theta}_k$	Trainable parameters of k^{th} layer
n	Number of samples in the dataset
y_i^t	True label/ground truth of the i^{th} data sample
δ_k	Tunable scaling parameter of k^{th} layer

$$e_{ik} = y_i^t - \boldsymbol{\theta}_k^T \mathbf{x}_i, \quad (3)$$

where e_{ik} is the error of k^{th} iteration on i^{th} data sample, y_i^t is the i^{th} truth label, x_i is i^{th} the data sample and $\boldsymbol{\theta}_k^T$ represents the trainable parameters during the k^{th} iteration. For the entire dataset, the vectorised form can be expressed as:

$$\mathbf{e}'_k = \mathbf{y}^t - \boldsymbol{\theta}_k^T \mathbf{X}, \quad (4)$$

where $\mathbf{e}'_k$, $\mathbf{y}^t$ are the vector forms of their respective variables and $\mathbf{X}$ represents the whole dataset as a matrix. However, the estimated error $\mathbf{e}'_k$ is imperfect as its value depends upon the size of the dataset. Hence, the error is further normalised as such:

$$\mathbf{e}_k = \left\| \frac{\mathbf{y}^t - \boldsymbol{\theta}_k^T \mathbf{X}}{n} \right\|, \quad (5)$$

where n is the number of total samples in the dataset. On substituting (5) in (2), we obtain:

$$\mathbf{y}_k = \Pi \left[\mathbf{y}_{k-1} - \delta_k \left\| \frac{\mathbf{y}^t - \boldsymbol{\theta}_k^T \mathbf{X}}{n} \right\| \right]. \quad (6)$$

The above expression represents an iterative algorithm to estimate $\mathbf{y}_k$. In the deep unfolding approach we unfold (6) into layers of a neural network and describe the resulting architecture in Section III.

III. PROPOSED PU-DETNET ARCHITECTURE

A. Proposed Architecture for PU-DetNet Scheme

The proposed architecture contains various components which are listed in Table. II.

TABLE II: Components of the architecture used in this paper

Component	Description
$\mathbf{W}_{1k}, \mathbf{b}_{1k},$ $\mathbf{W}_{2k}, \mathbf{b}_{2k},$ $\mathbf{W}_{3k}, \mathbf{b}_{3k}$	Trainable weights and biases of the k^{th} Layer
$\mathbf{z}_k$	Output of the k^{th} hidden layer
$\mathbf{v}_k$	Propagation vector output of the k^{th} layer
y_k	Output of the k^{th} layer
α	Output residual constant
h_s	Size of hidden layer
x_s	Number of columns in dataset / Size of a single example
n_l	Number of hidden layers
y_s	Size of the output (Default value = 1)
v_s	Size of the propagation vector
ReLU	Rectified linear unit function

From (6) it is clear that the architecture follows a relationship such that a layer receives the previous layer's output, reduce the error and make it closer to the ground truth. As mentioned, we utilise (6), implement the idea and unfold it to formulate the design of a single layer of the proposed PU-DetNet. In case of a generic neural network, the output of any given layer is dependent on just the previous layer's output (or just the input data, in case of the first hidden layer). However, from (6) y_k can be seen as a linear combination of y_{k-1} and $\mathbf{x}$. Thus, the output of k^{th} hidden layer can be given as:

$$\mathbf{z}_k = \text{ReLU} \left(\mathbf{W}_{1k} \begin{bmatrix} \mathbf{x} \\ \mathbf{v}_{k-1} \\ y_{k-1} \end{bmatrix} + \mathbf{b}_{1k} \right) \quad (7)$$

$$\mathbf{v} = \mathbf{W}_{2k} \mathbf{z}_k + \mathbf{b}_{2k} \quad (8)$$

$$y = \mathbf{W}_{3k} \mathbf{z}_k + \mathbf{b}_{3k} \quad (9)$$

$$\mathbf{v}_k = \alpha \cdot \mathbf{v} + (1 - \alpha) \mathbf{v}_{k-1} \quad (10)$$

$$y_k = \alpha \cdot \text{ReLU}(y) + (1 - \alpha) y_{k-1} \quad (11)$$

Fig. 2: Architecture of a single layer of PU-DetNet

Fig. 3: Nodal structure of a single layer PU-DetNet

Fig. 2 and expressions (7) - (11) describe a layer of the proposed PU-DetNet architecture, where the dashed box represents a single layer of the architecture. Every layer has a unique set of trainable weights and biases.

Fig. 3 describes the structural relationship between the input nodes, the hidden nodes, and the output nodes of a single layer of PU-DetNet. Fig. 3 (a) describes the process of obtaining the propagation vector v_k of a given layer while Fig. 3 (b) describes how a layer obtains the estimated classification y_k . As seen in both the figures, the input sequence of nodes receives

as its input the spectrum dataset ($\mathbf{x}$), the predicted output provided by the previous layer, and the propagation vector of the previous layer. The architecture uses a sequence of hidden nodes $\mathbf{z}_k$ (with parameters $\mathbf{W}_{1k}, \mathbf{b}_{1k}$) to expand upon the knowledge provided by the input nodes. Furthermore, the propagation vector $\mathbf{v}_k$ is obtained that passes on the inference obtained by layer k to layer $k + 1$. After obtaining $\mathbf{v}_k$, the architecture obtains the layer's estimated output of y_k which is a singular scalar value that estimates the probability of PU's presence. From Fig. 2 and Fig. 3, it is understood that the intermediate outputs of each layer $\mathbf{v}_k$ and y_k are obtained via different sets of weights and biases: $(\mathbf{W}_{2k}, \mathbf{b}_{2k})$ and $(\mathbf{W}_{3k}, \mathbf{b}_{3k})$ respectively. The output y_k of the k^{th} layer is also used to calculate the loss value of layer k which further contributes to the aggregate loss function as described in the following subsection.

B. Description of the Loss Function

In the process of training learning systems, a loss function is typically used with gradient descent to update trainable parameters of the model, such that the value of the loss function reduces in successive epochs of the training process. Since the problem under consideration is binary classification (i.e., whether PU is present or not), the intuitive choice of the loss function is binary cross entropy. The binary cross entropy between the ground truth and the architecture output for one data sample is described as [43]:

$$l_{nk} = y_n^t \cdot \log(P(y_n)_k) + (1 - y_n^t) \cdot \log(1 - P(y_n)_k), \quad (12)$$

where y_n^t is the ground truth of n^{th} data sample and $P(y_n^t)_k$ is the probability of y_n^t being true i.e., the estimated output of the k^{th} layer of architecture.

During the training part, learning architectures calculate the loss function of a specific portion of the dataset at once. In case of batch gradient descent, the architecture calculates the loss value for the entire dataset as follows:

$$l_k = \frac{1}{N} \cdot \sum_{n=1}^N l_{nk} = \frac{1}{N} \cdot \sum_{n=1}^N y_n^t \cdot \log(P(y_n)_k) + (1 - y_n^t) \cdot \log(1 - P(y_n)_k). \quad (13)$$

Unlike neural networks where the loss function is calculated using the output of just the final layer, PU-DetNet is designed such that it accounts for the loss of *every single layer* during the training process. Therefore, we use the following loss function which combines the losses of all layers as follows:

$$l = \frac{1}{K} \cdot \sum_{k=1}^K l_k = \frac{1}{K} \cdot \frac{1}{N} \cdot \sum_{k=1}^K \sum_{n=1}^N y_n^t \cdot \log(P(y_n)_k) + (1 - y_n^t) \cdot \log(1 - P(y_n)_k) \quad (14)$$

The proposed PU-DetNet architecture aims to learn the sophisticated underlying structure of the PU detection problem in a computationally efficient manner, without compromising the accuracy of estimation. Hence, the architecture is designed to be deep in layers, such that it can capture the complicated patterns of a highly noisy dataset. One common problem of deep architectures is increased computational complexity. To overcome the same, we incorporate the concept of loss binding where layers of the architecture are bound in such a way that their outputs and hence their losses are tied together and thus behave similarly in terms of estimation. In implementing the same, we add the losses of all layers to obtain an aggregate loss of the whole architecture. This aggregate loss is further used by gradient descent like optimization methods.

This idea was inspired by the concept of auxiliary classifiers presented in GoogleNet [44] that aimed to encourage discrimination in the lower stages in the network such that intermediate layers would be used directly as classifiers. Due to this, a non conventional design of the loss function was described in (14). Due to loss binding, each layer may act as a classifier of its own, and hence have a loss function value.

The effect of binding the loss function of each layer is such that every layer is constrained to be optimised and be equally close to the ground truth. This effect further leads to infer that even though the training is performed on a deep architecture, one can use outputs from even

the shallower layers of the model. In implementing such an idea, where only shallower layers are required while testing, we observe the need for a high number of layers during training. Theoretically, the higher number of training layers can help build a more descriptive network as the number of trainable parameters increases. Due to loss binding, the layers succeeding any given layer form a backward feedback system as the optimisation of the deeper layers can affect the optimisation of preceding layers as well. The evidence and observation regarding this idea is provided in Section V.

C. Processing the Architecture

In this section, we define the training and testing of the proposed PU-DetNet architecture.

1) *Training Phase*: The parameters of the architecture are randomly initialized and iteratively updated for a number of times. The detailed process of training the PU-DetNet architecture is described in **Algorithm 1**. The function EstimateOutput() is based on equations (7) - (11), while the function CalculateBindedLoss() is based on (14). The function UpdateParameters() is an optimization function and in the case of PU-DetNet, it is chosen to be Adam Optimizer [45] due to its properties like lower computational complexity and lower dependence on hyperparameter tuning.

Algorithm 1 Training the proposed PU-DetNet Scheme

Input: $X_{\text{train}}, y_{\text{train}}, \text{epochs}$

Output: Parameters

```

1: Parameters  $\leftarrow$  RandomInitialization()
2: for  $i \leftarrow 1$  to epochs do
3:    $y_{\text{est}} \leftarrow \text{EstimateOutput}(X_{\text{train}}, \text{Parameters})$ 
4:    $\text{loss} \leftarrow \text{CalculateBindedLoss}(y_{\text{est}}, y_{\text{train}})$ 
5:   Parameters  $\leftarrow \text{UpdateParameters}(\text{loss}, \text{Parameters})$ 
6: end for
7: return Parameters

```

2) *Testing Phase*: The optimal parameters obtained after the process of training the model collectively represent the final architecture which is ready to be tested. While testing, the number of correct and incorrect estimations (classifications) are recorded and the performance metrics

Algorithm 2 Testing the proposed PU-DetNet scheme

Input: $X_{\text{test}}, y_{\text{test}}, \text{Parameters}^*$
Output: P_d, P_f

```

1:  $N \leftarrow \text{length}(X_{\text{test}})$ 
2:  $\text{Out}_{\text{incorr}} \leftarrow 0$ 
3:  $\text{Out}_{\text{corr}} \leftarrow 0$ 
4: for  $i \leftarrow 1$  to  $N$  do
5:    $y_{\text{est}} \leftarrow \text{EstimateOutput}(X_{\text{test}}(i), \text{Parameters})$ 
6:   if  $y_{\text{est}} = 1$  and  $y_{\text{test}}(i) = 1$  then
7:      $\text{Out}_{\text{corr}} \leftarrow \text{Out}_{\text{corr}} + 1$ 
8:   else if  $y_{\text{est}} = 1$  and  $y_{\text{test}}(i) = 0$  then
9:      $\text{Out}_{\text{incorr}} \leftarrow \text{Out}_{\text{incorr}} + 1$ 
10:  end if
11: end for
12:  $P_d \leftarrow \frac{\text{Out}_{\text{corr}}}{N}$ 
13:  $P_f \leftarrow \frac{\text{Out}_{\text{incorr}}}{N}$ 
14: return  $P_d, P_f$ 

```

probability of detection (P_d) and probability of false alarm (P_f) are calculated. The detailed process of testing the PU-DetNet architecture is described in **Algorithm 2**. To evaluate the model's ability to classify the input signal correctly, we utilise the performance metrics: P_d and P_f as described in (15) and (16) respectively.

$$P_d = \frac{\text{No. of correctly classified } \mathcal{H}_1 \text{ samples}}{\text{Total no. of } \mathcal{H}_1 \text{ samples fed}} \quad (15)$$

$$P_f = \frac{\text{No. of incorrectly classified } \mathcal{H}_0 \text{ samples}}{\text{Total no. of } \mathcal{H}_0 \text{ samples fed}} \quad (16)$$

Furthermore, in DSA/CR systems, spectrum sensing has direct impact on throughput. The relation between sensing time (T_s) and the throughput for SU can be expressed as:

$$\text{Throughput} = \frac{(T - T_s)}{T} \times B \cdot \log_2(1 + \text{SNR}), \quad (17)$$

where T is the frame duration in DSA/CR networks, and B is the bandwidth. These metrics are further used in Section V to demonstrate experimental results.

IV. CONSIDERED DATASETS AND PREPROCESSING

Algorithm 3 Processing the signal data**Input:** X_{Signal} , n **Output:** X_{Energies} , labels

```

1: data_size  $\leftarrow \frac{\text{length}(X_{\text{Signal}})}{n}$ 
2: for SNR  $\leftarrow -20\text{dB}$  to  $4\text{dB}$  do
3:    $X_{\text{Noisy}} \leftarrow X_{\text{Signal}} + \text{AWGN}$ 
4:   for  $i \leftarrow 1$  to data_size do
5:      $X_{\text{Energies}} \leftarrow \text{Append}(\text{Energy}(X_{\text{Noisy}}, i, \text{data\_size}))$ 
6:     labels  $\leftarrow \text{Append}(1)$ 
7:      $X_{\text{Energies}} \leftarrow \text{Append}(\text{Energy}(\text{AWGN}, i, \text{data\_size}))$ 
8:     labels  $\leftarrow \text{Append}(0)$ 
9:   end for
10: end for
11: return  $X_{\text{Energies}}$ , labels

```

To validate the proposed scheme, we have considered data from five different datasets corresponding to signal formats whose detection performance is relevant in several realistic spectrum sharing and coexistence scenarios. These datasets were processed in such a manner that energy values of fixed time sensing events were taken into account. As described in **Algorithm 3**, all the considered datasets were transformed to make them suitable for validating the performance of a scheme in the task of PU detection. These processed datasets were further used to compare the performance of the proposed PU-DetNet scheme with current state-of-the-art. The results of the same will be shown in Section V.

A. Empirical Test-Bed Setup

We deployed an empirical test-bed setup on the roof-top of the School of Engineering and Applied Science, Ahmedabad University for spectrum data acquisition, the details of which are reported in our previous works [20], [29], [30], omitted here for the sake of brevity. The empirical measurement setup is shown in Fig. 4. The aforementioned setup was used to capture raw signal data of four bands: FM, GSM, DCS, and UHF. The specifications of signal data captured are described in Table III. This dataset contains more than 1,000,000 samples per band per SNR.

Fig. 4: Empirical measurement test-bed setup [29]

TABLE III: Channels measured in empirical setup and USRP configuration [29].

Radio Technology	Channel Number	f_{start} (MHz)	f_{center} (MHz)	f_{stop} (MHz)	Signal Bandwidth (MHz)	Decimation Rate	Sampled Bandwidth (MHz)
FM broadcasting	—	96.500	96.700	96.900	0.2	64	1
E-GSM 900 DL	77	950.2	950.4	950.6	0.2	64	1
DCS 1800 DL	690	1839.6	1840.8	1841	0.2	64	1
UHF television (Band IV)	U-33	566	570	574	8	8	8

B. 5G Simulated Dataset

The detection of 5G signals is relevant in spectrum sharing scenarios such as those enabled by the 5G NR-U technology, where the presence of 5G NR waveforms in unlicensed bands needs to be detected. We used MATLAB 5G toolbox in generating 5G waveforms which are compliant with 3GPP Release 15 [46]. Test models from the waveform generator were used to obtain signal data from four different configurations, each having a different set of parameters as shown in Table IV. This dataset contains 153,600 samples per SNR for each configuration.

TABLE IV: Parameters of the configurations of the 5G-NR test models for data simulation

	Config. A	Config. B	Config. C	Config. D
Frequency Range	450 MHz - 6 GHz	450 MHz - 6 GHz	24.25 GHz - 52.6 GHz	24.25 GHz - 52.6 GHz
Modulation	64QAM	QPSK	QPSK	64QAM
Channel Bandwidth	10 MHz	10 MHz	50 MHz	50 MHz
Subcarrier Spacing	60 kHz			
Duplex Mode	Frequency Division Duplex (FDD)			

TABLE V: Parameters of the DVB-S2 simulation

Parameter	Value
Samples per symbol	2
Rolloff Factor	0.35
Channel Bandwidth	3.6×10^7
Carrier Frequency Offset	3×10^6
Sampling Clock Offset	5 ppm

TABLE VI: Parameters of the simulated radar waveform

Parameter	Config. A	Config. B
Pulses per second	2130	2340
Pulse width	5×10^{-6} m	
Pulses per burst	20	14
Chirp Width	8×10^7 m	6×10^7 m
Chirp Direction	Down	Up
Sampling Frequency	10^7 Hz	

C. DeepSig Dataset

The publicly available DeepSig dataset [47] (RADIOML 2016.10A) contains signal data consisting of 11 modulations (8 digital, 3 analog). This dataset was first released at the 6th annual GNU radio conference and is useful in the context of this work to assess the performance of the proposed PU-DetNet scheme with commonly used signal modulations. While typically used for modulation classification, we utilise this dataset for PU-detection after processing it as described in Algorithm 3. It should be noted that unlike other mentioned datasets, no noise was added to the DeepSig dataset considering that the signal data were already presented with SNR values from -20dB to 18dB. However for training and testing, signal data with SNR values from -20dB to 4dB from four modulations were considered. Out of the 11 modulations, signal data of the modulations BPSK, QPSK, 64-QAM and GFSK were considered and processed. There are 128,000 samples per SNR, per modulation scheme in this dataset.

D. Satellite Communications Dataset

CR has been proposed to enable spectrum sharing not only in terrestrial but also in satellite communication bands and therefore the performance of spectrum sensing methods with satellite communication signals is relevant as well. We utilized MATLAB's satellite communications toolbox [48] that provides standards-based tools for designing, simulating, and verifying satellite communications systems and links. To obtain the dataset, an end-to-end DVB-S2 simulation with RF impairments and corrections was used. The parameters of the configuration are described in Table V. This dataset contains 800,000 samples per SNR.

E. Radar Dataset

The software tool [49] provides a radio frequency (RF) dataset generator for incumbent signals in the 3.5 GHz citizens broadband radio service (CBRS) Band, which is another practical scenario where spectrum sensing is relevant. The pulse modulation types for the radar signals and their parameters are selected based on national telecommunications and information administration (NTIA) testing procedures for environmental sensing capability (ECS) certification. Using the generator provided, two simulated radar waveforms with varying parameters were obtained to validate the PU-DetNet scheme. The parameters of the configurations set up to obtain these waveforms are described in Table VI. This dataset contains 800,000 samples per SNR for each configuration.

V. EXPERIMENTAL RESULTS

In this section, we describe the experiments comparing the proposed PU-DetNet scheme and other state-of-the-art ML based sensing schemes. The training and testing of the proposed PU-DetNet architecture were performed with the aid of the Tensorflow library [50]. For processing the proposed architecture and benchmark architectures, we used a 12GB NVIDIA Tesla K80 GPU offered by Google Colab. After training the PU-DetNet scheme on train dataset and obtaining the optimal parameters, the scheme was evaluated on the test dataset which comprise of approximately 30% of the considered dataset.

A. Model Evaluation

In learning architectures, performance is often evaluated by the value of loss function which describes the behaviour of the model with respect to the ground truth. The loss function of the PU-DetNet architecture is described in (14). This loss is calculated and then plotted with respect to epochs i.e., the number of times a dataset is passed through the architecture during the training phase. It can be observed in Fig. 5 (a) that the train and test loss decrease with the number of epochs. An elbow-shaped curve is obtained with a cut-off at around 100 epochs. This states

Fig. 5: Behaviour of loss function for FM band with SNR = -5 dB

TABLE VII: Variation in P_d with respect to number of layers used for training (columns) and number of layers used for testing (rows) at SNR = -4dB, Empirical test bed setup, FM band

Train Test	2	5	15	25	50	100
2	0.6484	0.6993	0.7293	0.7620	0.8729	0.9256
5	-	0.7564	0.7864	0.8252	0.9443	0.9755
15	-	-	0.8379	0.8704	0.8570	0.8961
25	-	-	-	0.8406	0.8532	0.8530
50	-	-	-	-	0.7958	0.8183
100	-	-	-	-	-	0.8002

that the loss reduction slows down after 100 epochs. The reduction in loss over epochs suggests that with increasing training, the model performs better on data. It can also be seen that the loss over test data remains slightly higher than the loss over train data which is quite intuitive given the fact that test data were unseen by the model when this result was obtained. Fig. 5 (b) demonstrates the PU-DetNet's property of loss binding as described in Section III. It can be observed that the loss value quickly drops and almost becomes constant after layer 2. Thus, we can infer that after the architecture is completely trained on the dataset, the losses across the layers are bound to have similar values. This idea suggests that in a completely optimized PU-DetNet, the estimates of the shallower layers are as close to ground truth as the deeper layers of the architecture. However, we would like to highlight that all the layers are necessary

as confirmed through Table VII.

Table VII shows the value of P_d obtained from a model where for a given value in the table the column index represents the number of layers present in the architecture while training and the row index represents the number of layers used for testing. The value indicated in boldface represents the optimum value of P_d for a given number of training layers. We can notice that for a model with 100 training layers, the optimal output is obtained at shallower layer (layer 5 in our case) and it is assumed that the succeeding layers overfit the structure of the dataset. The need for higher number of layers while training is clearly observed in Table VII. When 50 or 100 layers are used, the optimum output is provided by the 5th layer, however this doesn't mean that the network could operate at the same level of performance with a lower number of layers for training, since in the case of a network with only 5 layers the accuracy is significantly lower than in the case of 100 layers. This is because feedback across all layers helps optimise the performance of shallower layers, even if the best output is not necessarily seen in the very last layer. As the number of training layers decrease, the overfitting decreases and the optimum shifts towards more number of testing layers. The training layers that form a backward feedback system with the aid of loss binding are not observed to be necessary in making a decision regarding the presence of PU. Thus, the layers succeeding the layer that was used while testing, were discarded and a computationally efficient system was formed. It can also be seen that even if the number of testing layers are to be kept constant, increasing the number of training layers helps in detecting the presence of PU as the value of P_d increases with increase in training layers.

Fig. 6 shows the accuracy of the PU-DetNet architecture in classifying correctly over train and test data. An observation from this figure can be made that the accuracies remain at 50% till around 350 epochs. It can be inferred that the model does not learn much about the underlying structure of the data and hence displays a random behaviour on this binary labelled data. However, after around 350 epochs the accuracies spike and the model starts learning the underlying structure

Fig. 6: Comparison of train and test accuracies with respect to epochs for (Empirical Test bed dataset, FM broadcasting band, SNR = -5dB)

Fig. 7: P_d vs. SNR for the proposed PU-DetNet scheme at various values of P_f (Empirical test bed dataset, FM broadcasting band)

TABLE VIII: Hyperparameters of the considered schemes

Hyperparameter	PU-DetNet	LSTM	CNN	ANN			
Epochs Trained	1000	20	50	40			
Layers Trained	90	2	2	4			
Tested on Layer No. (n_l)	2	2	2	4			
Nodes per Hidden Layer (h_s)	7	3	30, 5	8			
Size of the Vector Output (v_s)	4	N.A.	N.A.	N.A.			
Loss function	Binded BCE	Binary Cross Entropy (BCE)					
Optimiser	Adam Optimiser						
Activation function	ReLU						

of the dataset as the training accuracy immediately rises up to 85% and the test accuracy rises up to 65%. With increasing epochs, the gap between train and test accuracy slowly decreases as they converge. This indicates that with increasing epochs, the model's ability to generalize improves on unseen data. Furthermore, we observe the probability of the model to detect the presence of PU in spectrum with respect to the SNR value for varying values of P_f . Fig. 7 displays the behaviour of the scheme with respect to different values of SNR and P_f .

B. Sensing Performance Comparison with State-of-the-art Architectures

The performance of PU-DetNet is compared with the state-of-the-art architectures in spectrum sensing: The LSTM based spectrum sensing scheme [29], CNN based spectrum sensing scheme

Fig. 8: Comparison and validation of detection probability on the empirical test-bed data set for the considered spectrum sensing schemes: proposed PU-DetNet, LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] ($P_f = 0.05$)

[25] and the ANN based spectrum sensing scheme [20]. For fair comparison, all the three architectures were trained, tested on same datasets. Hyperparameters of these architectures were tuned to ensure optimum results. After tuning, the hyperparameters finalised for obtaining results are as seen in Table VIII.

Fig. 8 shows the comparison of P_d between the three schemes on empirical test bed dataset. An average gain of 42.04%, 57.42% and 78.03% is observed at -10dB with respect to LSTM, CNN and ANN schemes respectively on this dataset. Fig. 9 further makes the comparison with respect to the 5G simulated dataset and an average gain of 47.43%, 63.74% and 86.11% is observed at

(a) 5G NR test model with configuration A

(b) 5G NR test model with configuration B

(c) 5G NR test model with configuration C

(d) 5G NR test model with configuration D

Fig. 9: Comparison and validation of detection probability on 5G simulated Dataset for the considered spectrum sensing schemes: proposed PU-DetNet, LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] ($P_f = 0.05$)

-10dB with respect to LSTM, CNN and ANN schemes respectively. Fig. 10 validates the PU-DetNet scheme on the DeepSig dataset and an average gain of 56.03%, 84.66 %and 128.32% is observed at -10dB with respect to LSTM, CNN and ANN schemes respectively. Fig. 11 shows this comparison with respect to the satellite communications dataset and an average gain of 39.21%, 45.38% and 53.37% is observed at -10dB with respect to LSTM, CNN and ANN schemes respectively. Furthermore, Fig. 12 shows the comparison with respect to the radar dataset and an average gain of 40.06%, 58.50% and 63.86% is observed at -10dB with respect to LSTM, CNN and ANN schemes respectively. The proposed scheme consistently outperforms

Fig. 10: Comparison and validation of detection probability on DeepSig dataset for the considered spectrum sensing schemes: proposed PU-DetNet, LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] ($P_f = 0.05$)

the benchmarks scheme in terms of P_d . Although varying with different datasets, it can be said that the proposed DetNet scheme can yield an acceptable value of P_d ($P_d = 0.9$) at 2dB to 6dB of SNR lesser than the state-of-the-art schemes. Fig. 13 compares the receiver operating characteristic (ROC) curves of the three schemes. It can be observed that for a given P_f , the PU-DetNet scheme yields a higher P_d than the other two schemes. It is also observed that for lower values of SNR, all three schemes yield a lower P_d for a given P_f but the proposed method still outperforms the state-of-the-art methods LSTM, CNN and ANN.

Table IX shows the comparison of precision, recall and F1 score [51] observed for the proposed

Fig. 11: Comparison and validation of detection probability on satellite communications dataset for the considered spectrum sensing schemes: proposed PU-DetNet, LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] ($P_f = 0.05$)

(a) Simulated radar waveform with configuration A

(b) Simulated radar waveform with configuration B

Fig. 12: Comparison and validation of detection probability on radar dataset for the considered spectrum sensing schemes: proposed PU-DetNet, LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] ($P_f = 0.05$)

Fig. 13: ROC comparison of proposed Scheme with LSTM-based sensing [29], CNN-based sensing [25] and ANN-based sensing [20] at SNR = -10dB and SNR = -20dB (DeepSig Dataset, BPSK Modulation)

scheme and the benchmark schemes for data comprising of samples with SNR = -5 dB. Values marked in boldface represent the optimal value for each set of comparison. F1 score is commonly used to evaluate the accuracy of an algorithm and can be interpreted as a weighted average of the precision and recall. It can be appreciated from the table that the proposed PU-DetNet scheme outperforms the other state of the art sensing scheme in terms of F1 score.

TABLE IX: Precision, Recall and F1 score comparison of the proposed PU-DetNet scheme with the benchmark schemes on considered datasets for SNR = -5 dB

Dataset		Precision				Recall				F1 Score			
		ANN	CNN	LSTM	PU-DetNet	ANN	CNN	LSTM	PU-DetNet	ANN	CNN	LSTM	PU-DetNet
Empirical	FM	0.769	0.792	0.856	0.924	0.718	0.778	0.841	0.892	0.743	0.785	0.848	0.908
	E-GSM	0.791	0.814	0.873	0.917	0.744	0.794	0.864	0.913	0.767	0.804	0.868	0.915
	DCS	0.782	0.779	0.889	0.884	0.732	0.843	0.837	0.897	0.756	0.810	0.862	0.89
	UHF	0.723	0.836	0.844	0.941	0.773	0.771	0.879	0.865	0.747	0.802	0.861	0.901
5G	Config. A	0.764	0.823	0.861	0.935	0.711	0.789	0.840	0.928	0.737	0.806	0.850	0.931
	Config. B	0.734	0.791	0.886	0.872	0.776	0.769	0.845	0.893	0.754	0.780	0.865	0.882
	Config. C	0.744	0.829	0.839	0.906	0.724	0.841	0.838	0.880	0.734	0.835	0.838	0.893
	Config. D	0.783	0.819	0.879	0.921	0.765	0.844	0.839	0.909	0.774	0.831	0.859	0.915
DeepSig	BPSK	0.785	0.782	0.859	0.925	0.747	0.781	0.883	0.872	0.766	0.781	0.871	0.898
	QPSK	0.777	0.848	0.843	0.936	0.724	0.794	0.873	0.921	0.75	0.820	0.858	0.928
	GFSK	0.731	0.810	0.884	0.879	0.759	0.824	0.841	0.874	0.745	0.817	0.862	0.876
	QAM-64	0.769	0.844	0.840	0.912	0.744	0.783	0.887	0.885	0.756	0.812	0.863	0.898
Radar	Config. A	0.780	0.771	0.878	0.899	0.724	0.818	0.848	0.893	0.751	0.794	0.863	0.896
	Config. B	0.719	0.819	0.881	0.906	0.778	0.768	0.850	0.904	0.747	0.793	0.865	0.905
Sat. Comm.		0.788	0.822	0.849	0.883	0.747	0.791	0.844	0.891	0.767	0.806	0.846	0.887

C. Computational Complexity Analysis

Computational complexity is a necessary metric which must be considered when evaluating and comparing different schemes and their performance. The number of floating point operations (FLOPs) of is a measure of computational complexity that describes the total number of instructions a processor has to execute to perform the specific action. FLOP calculation can be done by studying the structure of a model and the final value depends largely on the hyperparameters of the model. Table X displays the formulae to calculate the total number of FLOPs taken up by a scheme in estimating the outputs. The hyperparameters that are considered while calculation are as described in Table II. In addition to the number of FLOPs, the time each scheme consumed for training and testing were also observed. As mentioned before, all three schemes were processed on a 12GB NVIDIA Tesla K80 GPU offered by Google Colab. Considering hyperparameters from Table VIII, it is intuitive that due to its deep nature the PU-DetNet scheme takes more time than ANN in training. However, PU-DetNet still manages to train completely in less time than LSTM as observed in Table X. Moreover, Table X describes the computational complexity analysis. It

TABLE X: Computational complexity analysis

Scheme	FLOPs Calculation	No. of FLOPs	Train Time (Whole Dataset)	Test Time (Single Sample)
PU-DetNet	$2 \cdot n_l((x_s + s_s + v_s) \cdot (h_s - 1))$	168	937.38 s	4.86 μs
LSTM	$8 \cdot 2 \cdot (x_s(h_s - 1) + n_l h_s(h_s - 1))$	448	1,397.32 s	58.51 μs
ANN	$2 \cdot (x_s(h_s - 1) + n_l h_s(h_s - 1))$	476	311.9 s	71 μs

Fig. 14: Throughput performance of the considered spectrum sensing methods: proposed PU-DetNet scheme, LSTM [29], and ANN-based sensing schemes [20] over various dataset.

can be observed that the proposed PU-DetNet scheme requires less number of FLOPs. Although the FLOPs account for the number of arithmetic operations undergone to perform a task, the actual time consumed by a scheme may vary depending upon biases, non linear activation functions and the complexity of the type of arithmetic/matrix operation. Note that although the FLOPs required for PU-DetNet is about 2.5 times lower than LSTM, the decreased amount of activations and biases, owing to the impact of loss binding has led to the computation time being an order of magnitude lower. The test time reduces by 91.69% and 93.15% over the LSTM and ANN schemes respectively. Hence PU-DetNet significantly outperforms the benchmark schemes LSTM and ANN, not only in terms of detection performance but also in terms of computational complexity.

D. Throughput analysis

In this section, we demonstrate the application of the proposed scheme on throughput. It is intuitive to note that the sensing time (time employed to sense PU) is analogous to the testing time of the ML architecture, as enlisted in the last column of Table X. Fig. 14 shows the plot of throughput v/s SNR validated over various dataset for the proposed PU-DetNet, LSTM and ANN based sensing schemes (with $T = 0.1\text{ms}$, and B as per the considered dataset). We can notice that the proposed PU-DetNet performs significantly better than the state of the art sensing schemes as on average gain of 87 % and 130 % can be observed as compared to LSTM scheme

and ANN scheme respectively at SNR = -5 dB. The gain in throughput is due to the fact that the test time for proposed PU-DetNet scheme is much less (and hence quicker detection) as compared to other scheme in Table X. This is inevitably due to the combined model based and data driven approach in the unfolded architecture.

VI. CONCLUSION

In this work, a deep unfolding approach is introduced for spectrum sensing problem that harvests the strength of both: model based and data driven approaches. The Primary User-Detection Network (PU-DetNet) is proposed to overcome the shortcomings of ML/DL frameworks like high computational complexity and black box like behaviour. A unique technique is described which involves binding the loss function across all layers that helps in reducing the computational complexity significantly. The proposed scheme is evaluated on five different datasets. The proposed scheme outperforms state-of-the-art data driven spectrum sensing schemes. Numerical results indicate that the proposed scheme reduces the test time by 91.69% and 93.15% compared to the LSTM and ANN based schemes respectively. Furthermore it was observed that at SNR = -10dB, probability of detection is improved by a significant amount compared to the LSTM approach (between 39% to 56%), CNN approach (45% to 84%) and the ANN approach (between 53% and 128%) using empirical, 5G simulated, DeepSig, satellite communications and radar dataset. Moreover, the proposed scheme also shows an improvement in throughput by 87% and 130% more as compared to LSTM and ANN based schemes respectively. This work provides a comprehensive study and offers a computationally efficient smart sensing framework for the devices with low computational abilities, as envisaged in future wireless networks.

REFERENCES

- [1] F. Liu, Y. Cui, C. Masouros, J. Xu, T. X. Han, Y. C. Eldar, and S. Buzzi, "Integrated sensing and communications: Towards dual-functional wireless networks for 6G and beyond," arxiv 2021.
- [2] H. Wymeersch and et. al, "Integration of communication and sensing in 6G: A joint industrial and academic perspective," arxiv 2021.
- [3] J. Lunden, V. Koivunen, and H. V. Poor, "Spectrum exploration and exploitation for cognitive radio: Recent advances," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 123–140, May 2015.

- [4] J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal," *IEEE Pers. Commun.*, vol. 6, no. 4, pp. 13–18, Aug. 1999.
- [5] X. Liu, Q. Sun, W. Lu, C. Wu, and H. Ding, "Big-data-based intelligent spectrum sensing for heterogeneous spectrum communications in 5G," *IEEE Wireless Commun.*, vol. 27, no. 5, pp. 67–73, Oct. 2020.
- [6] A. Kliks, L. Kulacz, P. Kryszkiewicz, H. Bogucka, M. Dryjanski, M. Isaksson, G. P. Koudouridis, and P. Tengkvist, "Beyond 5G: Big data processing for better spectrum utilization," *IEEE Veh. Technol. Mag.*, vol. 15, no. 3, pp. 40–50, Sep. 2020.
- [7] S. Haykin, "Cognitive radio: Brain-empowered wireless communications," *IEEE J. Sel. Areas Com.*, vol. 23, no. 2, pp. 201–220, Feb. 2005.
- [8] M. López-Benítez and F. Casadevall, "Modeling and simulation of time-correlation properties of spectrum use in cognitive radio," in *Proc. of CROWNCOM*, June 2011, pp. 326–330.
- [9] H. Urkowitz, "Energy detection of unknown deterministic signals," *Proc. IEEE*, vol. 55, no. 4, pp. 523–531, 1967.
- [10] F. F. Digham, M. S. Alouini, and M. K. Simon, "On the energy detection of unknown signals over fading channels," *IEEE Trans. Commun.*, vol. 55, no. 1, pp. 21–24, Jan. 2007.
- [11] M. López-Benítez and F. Casadevall, "Improved energy detection spectrum sensing for cognitive radio," *IET commun.*, vol. 6, p. 785 – 796, May 2012.
- [12] B. Soni, D. K. Patel, Z. Ding, Y. L. Guan, and S. Sun, "On sensing performance of multi-antenna mobile cognitive radio conditioned on primary user activity statistics," *IEEE Trans. Wireless Commun.*, pp. 1–1, 2021.
- [13] Z. Xu and J. Sun, "Model-driven deep-learning," *Natl. Sci. Rev.*, vol. 5, no. 1, pp. 22–24, 2018.
- [14] C. Jiang, H. Zhang, Y. Ren, Z. Han, K. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, Apr. 2017.
- [15] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.
- [16] S. Dörner, S. Cammerer, J. Hoydis, and S. T. Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.
- [17] J. Wang, C. Jiang, H. Zhang, Y. Ren, K. Chen, and L. Hanzo, "Thirty years of machine learning: The road to Pareto-optimal wireless networks," *IEEE Commun. Surveys Tuts.*, pp. 1–1, 2020.
- [18] J. Tian, Y. Pei, Y. Huang, and Y. Liang, "Modulation-constrained clustering approach to blind modulation classification for MIMO systems," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 4, pp. 894–907, Dec. 2018.
- [19] Y.-J. Tang, Q.-Y. Zhang, and W. Lin, "Artificial neural network based spectrum sensing method for cognitive radio," in *Proc. of IEEE WiCOM*, 2010, pp. 1–4.
- [20] M. R. Vyas, D. K. Patel, and M. López-Benítez, "Artificial neural network based hybrid spectrum sensing scheme for cognitive radio," in *Proc. of IEEE PIMRC*, Oct. 2017, pp. 1–7.
- [21] D. K. Patel, M. López-Benítez, B. Soni, and et al., "Artificial neural network design for improved spectrum sensing in cognitive radio," *Wireless Networks*, vol. 26, p. 6155–6174, Aug. 2020.
- [22] J. Tian, P. Cheng, Z. Chen, M. Li, H. Hu, Y. Li, and B. Vucetic, "A machine learning-enabled spectrum sensing method for OFDM systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11 374–11 378, Nov. 2019.
- [23] Z. Shi, W. Gao, S. Zhang, J. Liu, and N. Kato, "Machine learning-enabled cooperative spectrum sensing for non-orthogonal multiple access," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 5692–5702, 2020.
- [24] C. Liu, J. Wang, X. Liua, and Y. Liang, "Deep CM-CNN for spectrum sensing in cognitive radio," *IEEE J. Sel. Areas Commun.*, pp. 1–1, 2019.
- [25] C. Liu, X. Liu, and Y. Liang, "Deep CNN for spectrum sensing in cognitive radio," in *Proc. of IEEE ICC*, May 2019, pp. 1–6.
- [26] J. Xie, C. Liu, Y. Liang, and J. Fang, "Activity pattern aware spectrum sensing: A CNN-based deep learning approach," *IEEE Commun. Lett.*, vol. 23, no. 6, pp. 1025–1028, June 2019.

- [27] W. Lee, M. Kim, and D. Cho, "Deep cooperative sensing: Cooperative spectrum sensing based on convolutional neural networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 3005–3009, Mar. 2019.
- [28] Q. Cheng, Z. Shi, D. N. Nguyen, and E. Dutkiewicz, "Sensing OFDM signal: A deep learning approach," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7785–7798, 2019.
- [29] B. Soni, D. K. Patel, and M. Lopez-Benitez, "Long short-term memory based spectrum sensing scheme for cognitive radio using primary activity statistics," *IEEE Access*, vol. 8, pp. 97 437–97 451, 2020.
- [30] N. Balwani, D. K. Patel, B. Soni, and M. López-Benítez, "Long short-term memory based spectrum sensing scheme for cognitive radio," in *Proc. of IEEE PIMRC*, Sept. 2019, pp. 1–6.
- [31] J. Xie, J. Fang, C. Liu, and L. Yang, "Unsupervised deep spectrum sensing: A variational auto-encoder based approach," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5307–5319, 2020.
- [32] S. Zheng, S. Chen, P. Qi, H. Zhou, and X. Yang, "Spectrum sensing based on deep learning classification for cognitive radios," *China Communications*, vol. 17, no. 2, pp. 138–148, 2020.
- [33] J. Xie, J. Fang, C. Liu, and X. Li, "Deep learning-based spectrum sensing in cognitive radio: A CNN-LSTM approach," *IEEE Commun. Lett.*, vol. 24, no. 10, pp. 2196–2200, 2020.
- [34] M. Xu, Z. Yin, M. Wu, Z. Wu, Y. Zhao, and Z. Gao, "Spectrum sensing based on parallel CNN-LSTM Network," in *Proc. of IEEE VTC*, 2020, pp. 1–5.
- [35] A. Hoglund, X. Lin, O. Liberg, A. Behravan, E. A. Yavuz, M. Van Der Zee, Y. Sui, T. Tirronen, A. Ratilainen, and D. Eriksson, "Overview of 3GPP release 14 enhanced NB-IoT," *IEEE Network*, vol. 31, no. 6, pp. 16–22, 2017.
- [36] Q. Hu, Y. Cai, Q. Shi, K. Xu, G. Yu, and Z. Ding, "Iterative algorithm induced deep-unfolding neural networks: Precoding design for multiuser MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1394–1410, 2021.
- [37] J. R. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *arXiv preprint arXiv:1409.2574*, 2014.
- [38] A. Jagannath, J. Jagannath, and T. Melodia, "Redefining wireless communication for 6G: Signal processing meets deep learning," *arXiv preprint arXiv:2004.10715*, 2020.
- [39] M.-W. Un, M. Shao, W.-K. Ma, and P. C. Ching, "Deep MIMO detection using ADMM unfolding," in *IEEE Data Science Workshop (DSW)*, 2019, pp. 333–337.
- [40] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, 2019.
- [41] J. Zhang, Y. Li, Z. L. Yu, Z. Gu, Y. Cheng, and H. Gong, "Deep unfolding with weighted L2 minimization for compressive sensing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 3027–3041, 2021.
- [42] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Nanjing, China, Oct., 2019, pp. 266–271.
- [43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 1–9.
- [45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [46] "5G toolbox." [Online]. Available: <https://www.mathworks.com/products/5g.html>
- [47] T. J. O'shea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016.
- [48] "Satellite communications toolbox." [Online]. Available: <https://in.mathworks.com/products/satellite-communications.html>
- [49] M., Caromi, ror:05xpvk416, org:Organization, N. I. of Standards, Technology, Raied, foaf:Person, R. M. Caromi, R., Souryal, and et al. [Online]. Available: <https://data.nist.gov/od/id/mds2-2229>
- [50] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: <https://www.tensorflow.org/>
- [51] D. M. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," *arXiv preprint arXiv:2010.16061*, 2020.