JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

EFFECTIVE DATA TRANSMISSION USING SPECTRUM SENSING FOR WIRELESS COMMUNICATION

T. Himabindu[1**], UG Student, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru
T. Surya Teja[2], UG Student, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru
Sk. Dil Afroz[3], UG Student, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru
V. Bhavya[4], UG Student, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru
E. Vargil Vijay[5], Assistant Professor, Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru

***Abstract*—**This paper is about LSTM with ADAGRAD & ADAM optimizers which uses deep learning techniques to provide a unique method of spectrum sensing in cognitive radio networks. Conventional technique's performance is less because they are unable to adjust to changing radio frequency environments. Our approach enables real-time spectrum detection by LSTM using ADAGRAD Optimizer within the Jupyter Notebook environment of Python. We have evaluated its effectiveness against conventional methods by extensive experimentation with RF dataset, showing notable increases in accuracy and adaptability. Our results point to the potential of deep learning to reduce interference and scarcity of spectrum, improving network performance and spectrum usage. We also discuss about potential possibility for future study, such as examining scalable large-scale cognitive radio networks and advanced structures. In addition to advancing cognitive radio technology and spectrum sensing, this research has applications for next-generation wireless communication systems.

***Keywords*—** *Spectrum sensing, Deep Learning, LSTM.*

I. INTRODUCTION

Wireless networks rely heavily on spectrum sensing, especially in dynamic systems where devices compete for unused frequencies. This technique identifies vacant channels with enough bandwidth for data transmission. Spectrum sensing detects the presence or absence of primary users (licensed users) occupying a specific frequency band. Different approaches exist, each with varying complexity, accuracy, and computational needs. When spectrum resources are limited, spectrum sensing becomes even more crucial for reliable data transfer. Machine learning models are showing promise in surpassing traditional methods, and this work proposes a Convolutional Neural Network (CNN) model for spectrum sensing in environments with Gaussian noise [3]. This model is specifically designed to identify the presence of licensed users

Unlike conventional methods that often rely on predefined algorithms deep learning approaches can automatically learn relevant features from raw data. deep learning models, when properly trained, can exhibit robustness to noise and interference, enhancing their performance in challenging scenarios. Ultimately, the choice between deep learning and conventional methods depends on factors such as data availability, computational resources. Deep learning models may exhibit better performance in complex and dynamic spectrum environments due to their ability to capture complex patterns and dependencies in the data. it's important to note that deep learning techniques typically require a large amount of labeled data for training, and they may be relatively intensive, especially for real-time applications. In contrast, conventional methods may be simpler to implement and require fewer computational resources

This model development is done in the Python programming environment and made possible by the adaptable Jupyter Notebook platform, offers a favorable environment for the creation and improvement of advanced spectrum sensing algorithms. The system's thresholding methods and real-time spectrum sensing capabilities are designed to improve flexibility in dynamic radio settings and guarantee quick identification of available spectrum. In addition, the system includes capabilities for calculating and displaying performance data, which facilitates well-informed decision-making and effective spectrum use strategies.

II. Literature Survey

Among all the studies, recent literature explores methodologies integrating cognitive radio technology with

Long Short-Term Memory (LSTM) deep learning architecture and SELU/ELU optimizers to enhance spectrum sensing performance. These approaches combine feature extraction and deep learning to achieve superior results.

As the Signal-to-Noise Ratio (SNR) increases, the probability of detection (Pd) rises while the probability of false alarm (Pf) diminishes. Notably, Pd exceeds 80% and Pf is below 10% when SNR surpasses -10 dB [4]. Integrating Energy Detection (ED) with Error Vector Magnitude (EVM) further improves detection performance, with Pd increasing from 0.6 at SNR = -24 dB to over 0.95 at SNR = -12 dB, accompanied by a decrease in FAR [5]. These methods outperform traditional spectrum sensing techniques, particularly in low SNR conditions. The proposed CNN-LSTM spectrum sensing detector excels in capturing signal energy- correlation features and temporal Primary User (PU) activity patterns, highlighting superior performance compared to existing models such as CLDNN. At SNRs lower than -5 dB, it significantly reduces sensing errors, offering potential enhancements for cognitive radio networks [7].

III. BLOCK DIAGRAM

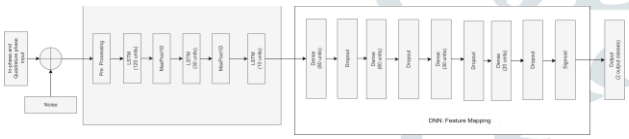


Fig.1 Proposed diagram based on LSTM

IV. PROPOSED METHODOLOGY

LSTM (Long Short-Term Memory)

In standard RNNs, the back propagation of gradients regularly meets the issue of diminishing gradients or explosively large gradients throughout training. This arises due to the frequent multiplication of weights over time, which can delay the network's ability to learn from long sequences of data successfully. To overcome this problem, LSTM units were enhanced with advanced gating mechanisms designed to regulate the flow of information within the network. These mechanisms include the Input gate, forget gate and Output gate individually controlled by separate units. These gates control how much data is stored, separated at each time stamp, letting the network recollect vital data over lengthy sequences.

Opposing traditional Recurrent Neural Networks (RNNs) with a single tanh layer in the hidden state, Long Short-Term Memory (LSTM) networks address the problem of vanishing gradients by introducing a gated cell structure. This cell includes combination of four layers that generate both the cell output and cell state, which are then approved on to the following LSTM cell. The key uniqueness in LSTM networks is their capability to selectively recollect information over time through gating mechanisms. These gates are realized using sigmoid and tanh activation functions to control the movement of information

Input Gate: The input gate picks what current information to store in the cell state. Regulates how much added information is extra to the cell state. It is thorough by the input gate unit, which deliberates the existing input, the preceding output, and the preceding cell state.

$$i_{nt} = \sigma\,(w_{in}\,[h_{nt-1}, x_{nt}] + b_{in}) \rightarrow (1)$$

Forget Gate: The forget gate selects what information to throw away from the cell state. Limits how much of the preceding cell State should be taken. It is measured by the forget gate unit, which contemplates the existing input and the preceding output.

$$f_{nt} = \sigma\,(W_{fg}[h_{nt-1}, x_{nt}] + b_{fg}) \rightarrow (2)$$

Output Gate: The output gate chooses what the next hidden state is. Directs how much of the cell state should be output. It is controlled by the output gate unit, which studies the present input, the earlier output, and the up-to-date cell state.

$$O_{nt} = \sigma\,(w_{og}[h_{nt-1}, x_{nt}] + b_{og}) \rightarrow (3)$$

Cell State Update: The cell state C_{nt} is updated by applying the forget gate f_{nt} to the preceding cell state and accumulation the outcome of the input gate i_{nt} applied to the original candidate values ascended by a *tanh* function.

$$C_{nt} = f_{nt} * C_{nt-1} + i_{nt} * \tilde{C}_{nt} \rightarrow (4)$$

andidate Cell state: the candidate cell state $\tilde{C}_{n\,t}$ is a midway calculation that signifies the new data that might be added to the cell state C_{nt} at given timestamp.

$$\tilde{C}_{n\,t} = \tanh\,(w_{cs}\,[h_{nt-1}, x_{nt}] + b_{cs}) \rightarrow (5)$$

Hidden State: The hidden state h_{nt} is the output of the LSTM cell at the present time step. It comprises information about the current input and the preceding hidden state.

$$h_{nt} = O_{nt} * \tanh(C_{nt}) \rightarrow (6)$$

Where, i_{nt} = Input Gate

f_{nt}= Forget Gate

O_{nt}= Output Gate

σ= Sigmoid function

W_{in},W_{fg}, W_{og}= Weights for respective gate neurons

h_{nt-1} = Output of previous LSTM Block

(At timestamp nt-1)

x_{nt}= Input at current timestamp

b_{in}, b_{fg}, b_{og}= Biases for respective gates

C_{nt}= Cell state at timestamp(nt)

$\tilde{C}_{n\,t}$ = Represents candidate for cell state at timestamp (nt)

h_{nt} = Hidden state at timestamp (nt)

w_{cs}= Weight of respective gate neuron

b_{cs}= bias for respective gate

V. METHODOLOGY

The key alteration differs in the gating mechanism. LSTM introduce three special sigmoid gates and one tanh layer. These gates act as smart filters, monitoring the flow of information through the cell. They regulate which information is related for the next cell and what can be rejected. The gate outputs range from 0 to 1, with 0 suggesting complete refusal and 1 representing full inclusion.

LSTMs force these gates to perform memory management. Information considered important is preserved by the cell, while the gates achieve its flow and prevent unrelated particulars from collecting over lengthy sequences. This gated memory architecture permits LSTMs to surpass tasks concerning long-range dependencies in serial data.

- At each time step nt, the input vector x_{nt}, signifies the spectrum dimensions or features take out from the received signal. This could contain parameters such as signal strength, bandwidth usage, etc.
- The hidden state h_{nt} of the LSTM captures the temporal dynamics and needs in the spectrum data. It characterizes the network's memory of past remarks and is reorganized at each time step based on the present input and preceding hidden state.
- The cell state C_{nt} supports the stable memory of the LSTM. It stores information about the spectrum patterns cultured over time and is updated using the input, forget, and output gates.
- The forget gate f_{nt} controls which information from the preceding cell state C_{nt-1} should be taken or rejected based on the current input □□□ and earlier hidden state h□□−1. It selects how much of the past spectrum information should be over and done.
- The input gate i_{nt} and candidate update $\tilde{C}_{n\,t}$ composed choose which new spectrum information should be combined into the updated cell state C_{nt}. The input gate controls the significance of the latest information, while the candidate update computes the new candidate values to be added to the cell state.
- The output gate O_{nt} regulates which information from the updated cell state C_{nt} should be used to produce the output h_{nt}. It controls the flow of information from the LSTM's internal memory to the output, which could be used for expecting future spectrum convention or making decisions about spectrum distribution.

The suggested model's operation is represented in the Fig. 1. The preprocessing unit receives the data set as input. Noise is additionally added to the preprocessing unit along with the dataset. Noise is brought to simulate interference in signal transmissions. The dataset is then split into two sections: smooth alerts and noisy indicators. This division allows thorough analysis of noise outcomes on sign processing techniques.

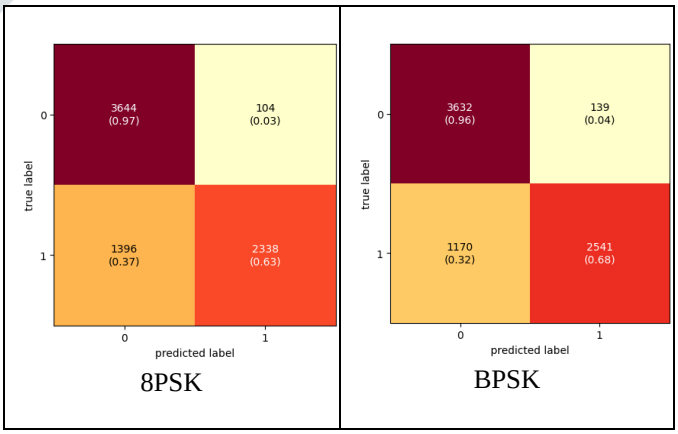
Our model architecture begins with an input layer shaped (128, 2) to accommodate signals. The data progresses through LSTM layers, the first having 120 units to capture complex temporal dependencies and returning sequences for further processing. A MaxPool1D layer with a pool size of 2 reduces temporal dimensionality. Subsequently, another LSTM layer with 10 units is employed for additional processing. GlobalMaxPooling1D diminishes temporal dimensionality, followed by dense layers with ReLU activation to extract features. Dropout layers with a dropout rate of 0.1 mitigate overfitting. Dense layers then gradually decrease feature dimensionality to 80, 60, 30, and 20 units respectively. Finally, a sigmoid activation and a single unit in the output layer perform binary classification, distinguishing between true and noise signals.

Binary cross-entropy loss is used to compute the difference among actual labels and predicted probabilities. The ADAGRAD and ADAM optimizers are utilized for parameter optimization, dynamically adjusting learning rates to enhance performance. Compiling the model with binary cross- entropy loss and ADAGRAD and ADAM optimizers guarantee effective discrimination between noisy and genuine indicators.

post-training, the version's performance is evaluated using assessment strategies to compute accuracy and loss metrics, supplying insights into generalization and effectiveness. These metrics are as compared towards thresholds or benchmarks to make decisions concerning model improvement or refinement

VI. RESULT

A classification model's performance in a dataset with binary labels is shown in Fig. 2, 3. For each modulation technique, such as BPSK, QPSK, 8PSK, and QAM 64 for both ADAGRAD and ADAM optimizers, the TP, TN, FP, and FN are represented by the confusion matrix.



8PSK BPSK

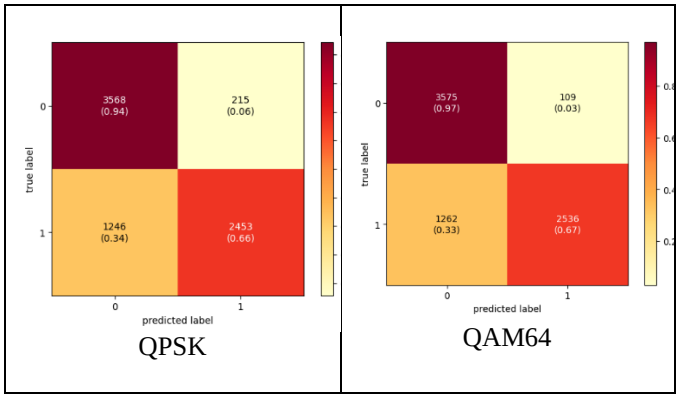


Fig.2 ADAGRAD Confusion Matrix for 8PSK, BPSK, QPSK and QAM 64

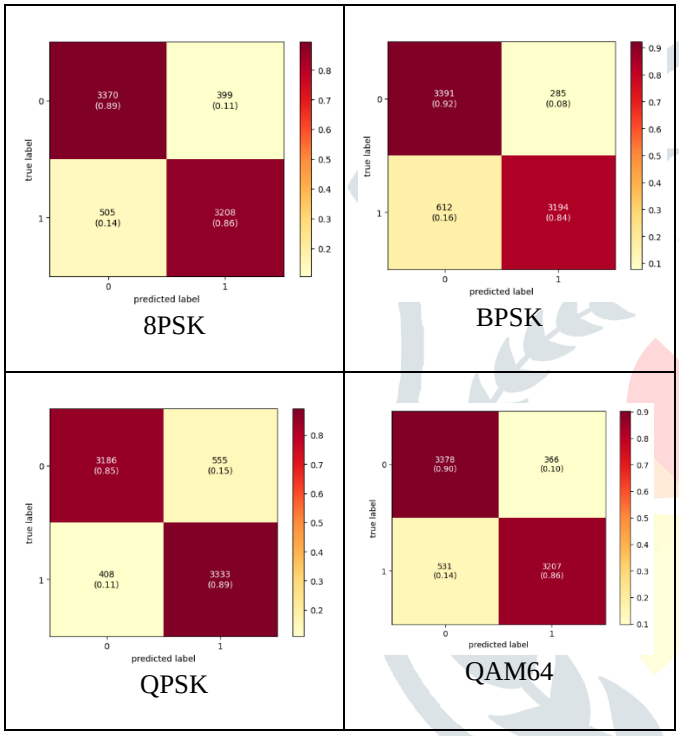


Fig.2 ADAM Confusion Matrix for 8PSK, BPSK, QPSK and QAM 64

Based on the TP, TN, FP, and FN values. The precision value for the modulation approaches was obtained by considering the SNR values within the range (-20 to 18 dB). This metric is often shown in Table 1, 2.

The accuracy values obtained indicate that, when compared to ADAM optimizer, ADAGRAD optimizer performs better in terms of precision under various modulation strategies.

Fig. 4 displays the accuracy versus SNR charts for the Adagrad optimizer. For modulation schemes such as BPSK, QPSK, 8PSK, and QAM64, the accuracy is measured at -6 dB SNR. The results show that BPSK, QPSK, 8PSK, and QAM64 have accuracy values of 0.82, 0.82, 0.81, and 0.83. This means that 82%, 82%, 81%, and 83% of the test sets samples are accurately predicted by these methods.

The accuracy versus SNR charts for the ADAM optimizer are shown in Fig. 5. The accuracy for modulation methods like QAM64, BPSK, QPSK, and 8PSK is assessed at -6 dB SNR. The accuracy values for BPSK, QPSK, 8PSK, and QAM64 are 0.91, 0.90, 0.81, and 0.90, according to the results. This indicates that these approaches accurately predict samples from 91%, 90%, 81%, and 90% of the test sets.

	QPSK	BPSK	8PSK	QAM64
-20	1	1	1	1
-16	1	1	1	1
-12	0.94	1	1	1
-10	0.96	0.93	0.95	1
-8	0.96	0.93	0.96	0.96
-6	0.97	0.93	0.97	0.95
-4	0.96	0.94	0.95	0.88
-2	0.89	0.90	0.95	0.36
0	0.95	0.98	0.96	0.96
2	0.95	0.94	0.96	0.96
4	0.86	0.88	0.89	0.91
6	0.90	0.98	0.95	0.95
8	0.88	0.93	0.94	0.94
10	0.94	0.96	0.95	0.96
12	0.84	0.96	0.97	0.97
14	0.96	0.98	0.98	0.94
16	0.85	0.89	0.91	0.92
18	0.93	0.97	0.98	0.98

Table.1 ADAGRAD Precision values

	QPSK	BPSK	8PSK	QAM64
-20	1	1	1	0.74
-16	0.83	0.83	0.83	0.70
-12	0.75	0.75	0.75	0.72
-10	0.76	0.76	0.76	0.62
-8	0.79	0.79	0.79	0.77
-6	0.89	0.89	0.89	0.89
-4	0.94	0.94	0.94	0.91
-2	0.96	0.96	0.96	0.97
0	0.99	0.99	0.99	0.97
2	0.98	0.98	0.98	0.99
4	0.99	0.99	0.99	1
6	0.98	0.98	0.98	1
8	0.99	0.99	0.99	0.99
10	0.98	0.98	0.98	0.94
12	0.99	0.99	0.99	1
14	0.99	0.99	0.99	0.99
16	0.99	0.99	0.99	1
18	0.98	0.98	0.98	1

Table.2 ADAM Precision values

By contrasting the accuracy of the Adam and Adagrad optimizers, it was found that Adam's accuracy is higher than that of the Adagrad optimizer.

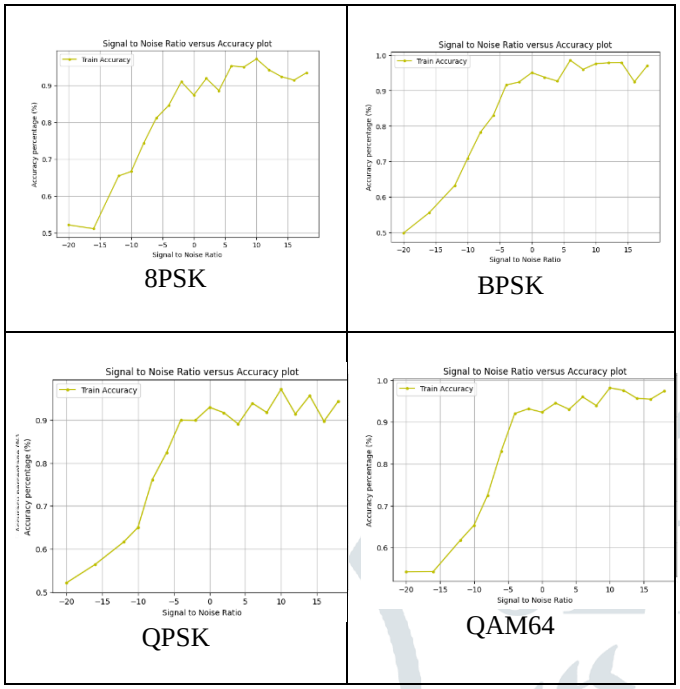


Fig. 4 ADAGRAD SNR V$_S$ Accuracy

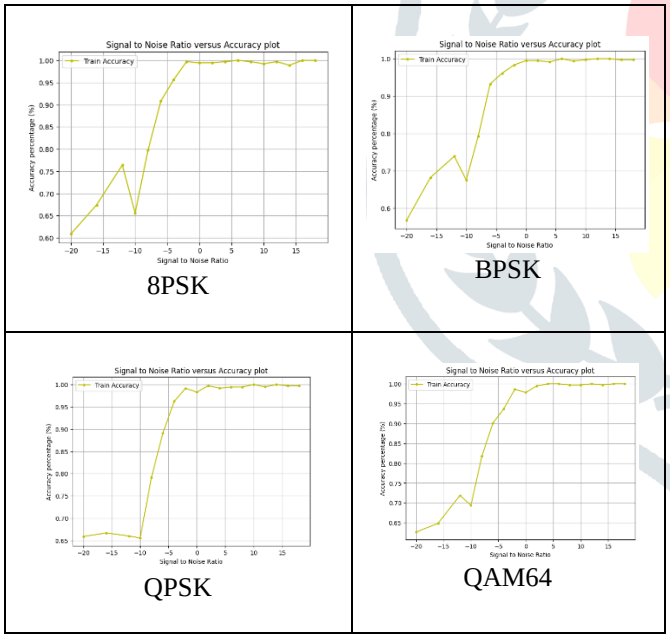
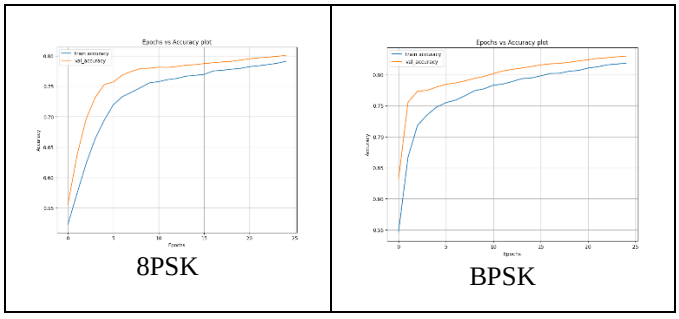


Fig .5 ADAM SNR VS ACCURACY



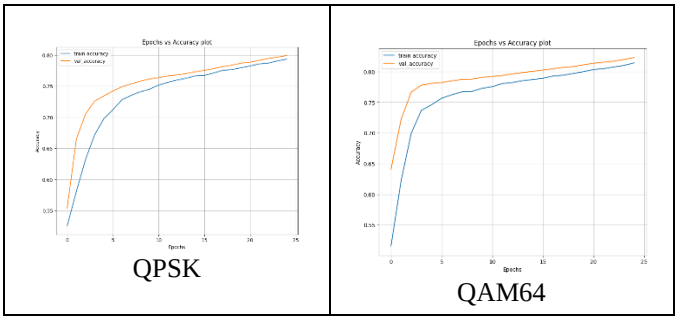


Fig.6 ADAGRAD Epochs v$_s$. Accuracy

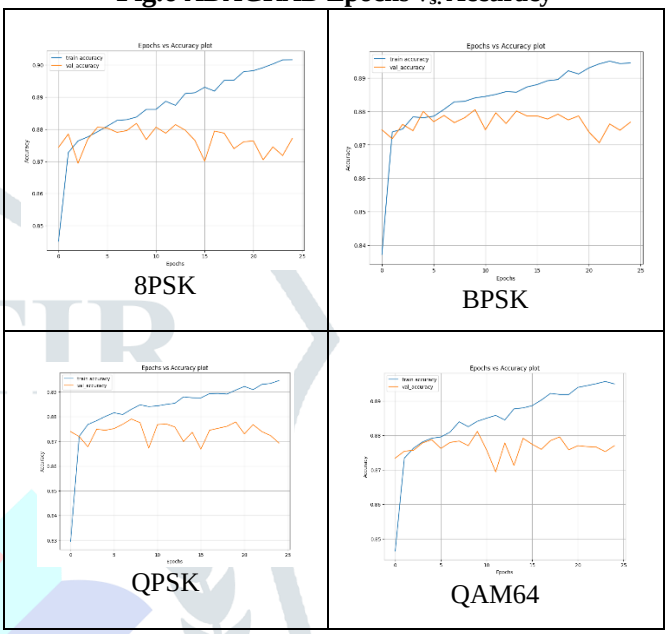


Fig. ADAM Epochs v$_s$. Accuracy

VII. CONCLUSION

Recently, there have been problems with spectrum sensing due to the increase of wireless network users compared to spectrum accessibility. The efficient assignment of the spectrum to users in this article was achieved through of deep learning models, which calls for a robust and reliable way to designate the spectrum as unused. To check whether a channel was free in this instance, the metrics TP, TN, FP, and FN were employed. The implementation of deep learning into spectrum sensing methodologies is a vital advancement in wireless communication. By overcoming the limitations of traditional approaches and offering a more adaptive and efficient solution, this article may contributes to the ongoing evolution of cognitive radio technology.

REFERENCES

[1]Felix Obite a, ∗ , Aliyu D. Usman b, Emmanuel Okafor. An overview of deep reinforcement learning for spectrum sensing in cognitive radio networks

[2] S. B. Goyal 1 & Pradeep Bedi 2 & Jugnesh Kumar 3 & Vijaykumar Varadarajan 4- Deep learning application for sensing available spectrum for cognitive radio: An ECRNN approach

[3] Anandakumar Haldorai, 1 Jeevanandham Sivaraj, 2

Munivenkatappa Nagabushanam, 3 and Michaelraj Kingston Roberts CognitiveWireless Networks Based Spectrum Sensing Strategies: A Comparative Analysis

[4] Geng, Yue, et al. "Spectrum sensing for cognitive radio based on feature extraction and deep learning." Journal of Physics: Conference Series. Vol. 2261. No. 1. IOP Publishing, 2022.

[5] Nasser, Abbass, et al. "A deep neural network model for hybrid spectrum sensing in cognitive radio." Wireless Personal 118.1 (2021): 281-299.

[6] Chen, Zhibo, et al. "*Deep STFT-CNN for spectrum sensing in cognitive radio*" in IEEE Communications Letters 25.3, 2020.

[7] Wang, Qian, et al. "*ConvLSTM based spectrum sensing at very low SNR.*" IEEE Wireless Communications Letters (2023).

[8]. LIUWEN LI 1, WEI XIE 2, AND XIN ZHOU 3 Cooperative Spectrum Sensing Based on LSTM-CNN Combination Network in Cognitive Radio System