



OPEN

## Optimized multi scale graph neural network with attention mechanism for cooperative spectrum sensing in cognitive radio networks

P. Nandhini<sup>1</sup>✉ & S. Vimalnath<sup>2</sup>

Cooperative spectrum sensing (CSS) plays a vital role in cognitive radio networks (CRNs). CSS enables efficient spectrum utilization and improves communication reliability through dynamic detection of underutilized frequency bands. However, the methods which evolved earlier face limitations like reduced detection accuracy and high false alarm rates in low Signal-to-Noise Ratio (SNR) conditions. Efficient allocation of spectrum resources in cognitive radio networks is affected by unreliable sensing at low signal-to-noise ratios, which leads to missed detections and false alarms. To overcome these challenges a novel optimized deep learning model is presented in this research work which provides reliable spectrum detection under noisy environments. The proposed Optimized Multi-Scale Graph Neural Network with Attention Mechanism (OMSGNNA) utilizes graph-based data representation and provides multi-scale feature extraction. The attention mechanism effectively fuses spatial and temporal information which enhances the performances. Additionally, an Adaptive Butterfly Optimization with Lévy Flights (ABO-LF) is incorporated to fine tune the parameters of proposed model. The proposed model experiments which are conducted using benchmark RadioML2016.10b dataset includes I/Q samples from 11 modulation schemes across SNR levels ranging from -20 dB to 18 dB. The performance evaluation exhibits better performance of proposed OMSGNNA in terms of 98% accuracy at high SNR with better precision, recall, and F1-score. Comparative analysis reveals that the proposed OMSGNNA outperforms traditional deep learning models even at low SNR conditions.

**Keywords** Cooperative spectrum sensing, Cognitive radio networks, Multi-scale graph neural network, Dual attention mechanism, Spatial-temporal fusion, Spectrum detection accuracy, Low signal-to-noise ratio, Intelligent spectrum utilization

Spectrum sensing (SS) in cognitive radio networks (CRNs) facilitates dynamic spectrum access by allowing more efficient use of underutilized frequency bands. It enables unlicensed secondary users to access the spectrum opportunistically while ensuring that their transmissions do not interfere with those of licensed primary users. Among various sensing techniques cooperative spectrum sensing (CSS) has developed as an effective approach due to its ability to aggregate observations from multiple secondary users. Due to this, CSS exhibits enhanced detection performance and reliability in complex noisy environments. The collaborative nature of CSS reduces the impacts of shadowing, fading, and other channel losses which are common in wireless communication scenarios<sup>1</sup>. Though CSS has potential benefits, realizing efficient and accurate detection remains challenging due to the trade-offs between detection performance, energy efficiency, and latency.

Another significant limitation of CSS in CRNs is its sensitivity to noisy environments particularly in low Signal-to-Noise Ratio (SNR) conditions. High noise levels lead to inaccurate detection. The noise factors increase false alarms or missed detections in the sensing process and affect the network operations. Additionally, the dependency on centralized architectures for data aggregation introduces scalability and energy efficiency challenges in the sensing process<sup>2</sup>. Since data fusion in CSS is computationally intensive and it introduces latency which further complicates real-time decision-making. Moreover, the presence of malicious secondary users affects the integrity of cooperative sensing which further redirects the decisions to incorrect spectrum status.

<sup>1</sup>Department of Electronics and Communication Engineering, V.S.B. Engineering College (Autonomous), Karur 639111, India. <sup>2</sup>Department of Electronics and Communication Engineering, M. Kumarasamy College of Engineering (Autonomous), Karur 639113, India. ✉email: nandhini0501phd@gmail.com

These limitations highlight the need for robust and scalable solutions that address the challenges associated with CSS<sup>3</sup>.

Conventional cognitive spectrum-sensing (CSS) schemes are largely rooted in energy detection, matched filtering, and cyclostationary feature analysis. Although these techniques are computationally lightweight, their reliability deteriorates when the signal-to-noise ratio is low or the signal characteristics are uncertain. To address these shortcomings, learning-driven methods have been introduced across wireless-network tasks and have delivered notable performance gains<sup>4–6</sup>. Classical machine-learning classifiers—such as Support Vector Machines (SVMs) and modern deep models like Convolutional Neural Networks (CNNs)—boost detection accuracy thanks to sophisticated feature-extraction and decision mechanisms<sup>7–18</sup>. Yet, they typically demand significant processing resources and struggle to model intricate spatial-temporal relationships present in spectrum data<sup>9–11</sup>. Heuristic optimizers, including Particle Swarm Optimization (PSO) and Genetic Algorithms (GA), have also been employed to locate vacant channels, but their sluggish convergence and limited adaptability inhibit real-time sensing. These limitations underscore the necessity for new frameworks that fuse the advantages of existing strategies while mitigating their respective drawbacks<sup>12</sup>.

The motivation of this research is to design a spectrum sensing model which is robust, scalable, and capable of operating effectively under varying SNR conditions. The primary objective is to enhance the detection accuracy in decision-making for cooperative spectrum sensing in CRNs. The proposed model aims to address the limitations by introducing a novel deep learning-based approach that integrates spatial and temporal features through advanced graph-based learning mechanisms. By focusing on adaptive optimization techniques and utilizing real-world datasets, the proposed model provides a practical and reliable solution for spectrum sensing in modern wireless communication systems.

An Optimized Multi-Scale Graph Neural Network with Attention Mechanism (OMSGNNA) is proposed in this research work as a novel approach for cooperative spectrum sensing. The proposed model utilizes graph-based representations to capture spatial dependencies among secondary users. The dual-level attention mechanisms integrate the features for enhanced feature fusion. A multi-scale pooling strategy is employed to aggregate local and global information effectively and the hybrid optimization algorithm ensures optimal parameter tuning. The novelty of this approach presents in its ability to utilize graph structures and attention mechanisms in attaining superior detection performance under challenging SNR conditions. The contributions made in the research work are summarized as follows.

- Proposed an Optimized Multi-Scale Graph Neural Network with Attention Mechanism (OMSGNNA) for cooperative spectrum sensing in cognitive radio networks. Spatial and temporal features are integrated through graph-based learning and dual-level attention mechanisms. The multi scale pooling strategy effectively aggregates the local and global information to attain enhanced spectrum sensing performances.
- Proposed a hybrid adaptive optimization algorithm for efficient hyperparameter tuning. The hybrid optimization model incorporates Adaptive Butterfly Optimization with Lévy Flights (ABO-LF) to fine tune the proposed model.
- Presented a detailed experimentation evaluation of the proposed model using benchmark RadioML2016.10b dataset across varying SNR levels to demonstrate superior detection performance.
- A scalable, robust, and computationally efficient mechanism with improved spectrum sensing accuracy has been proposed, which adapts dynamically to varying channel and environmental conditions. This enables enhanced spectrum utilization, reduces communication disruptions, and supports the development of intelligent wireless infrastructure for societal applications such as disaster recovery, smart cities, and rural connectivity.

## Related works and research gaps

The increasing demand for efficient spectrum utilization in wireless communication has driven significant advancements in spectrum sensing technologies, especially for CRNs. Traditional approaches often exhibit limitations such as noise uncertainty, lack of temporal correlation analysis, and inefficiencies under low SNR conditions. This literature review section presents a detailed analysis of existing methodologies and highlights the key contributions. The limitations in the existing sensing procedures in CRN are finally summarized to validate the research motivation.

### Related works

This section discusses some of the reviewed related works and the identified research gaps<sup>13–14</sup>. ML based CSS for CRN is presented in<sup>15</sup> to optimize spectrum utilization. The presented model includes a modified double-threshold energy detection scheme to reduce collision and false alarm probabilities to handle noise uncertainty. ML techniques like Gaussian Mixture Models (GMM), SVM, and KNN are used for energy vector classification. Additionally, a two-stage energy detector is employed to utilize adaptive thresholds which improves the detection accuracy. Simulation results highlight the presented approach efficiency over other classifiers in minimizing classification errors and maximizing detection probability under low SNR conditions. Though the presented model attained enhanced detection probability, its performance was limited by the lack of temporal correlation handling and dependency on energy-based thresholds, which further limits the model adaptability in rapidly changing environments.

A robust SVM-based ML presented in<sup>16</sup> incorporates SVM algorithm to classify CR-IoT users into normal and malicious categories using energy vectors as feature inputs. The algorithm effectively separates user groups by defining optimal hyperplanes to address the malicious behaviors. Experimental analysis exhibits the system effectiveness through high detection accuracy and improved reliability in spectrum sensing. The SVM's ability to handle non-linear separable data is validated through comparisons with different kernels and attains superior

classification accuracy. However increased latency and limited reliability are the limitations of the presented model. The ML based spectrum sensing model presented in<sup>17</sup> employs a Generalized Likelihood Ratio Test (GLRT) for initial spectrum sensing at secondary user devices. Further feature extraction is performed considering the parameters such as power spectral density, SNR, and signal energy. Finally, an intelligent fusion center that includes SVM as a classifier to process the features and exhibits better performance under low SNR conditions. Improved detection probability and spectrum utilization are the merits of the presented model. However, the presented model requires accurate parameter estimation and lags in performance when employed over diverse sensing environments.

The CSS model for CRN presented in<sup>18</sup> provides an Extreme Learning Machine that includes a feedforward neural network with a single hidden layer to attain better detection performances. The presented model exhibits reduced computational complexity over other learning machines by optimizing output weights and randomly initializing hidden layer parameters. Performance evaluations of the presented approach exhibit better accuracy in AWGN and Rayleigh fading environments compared to traditional ML algorithms like SVM and K-means clustering. Improved detection rates and reduced energy consumption are the observed features of the presented model. However, the limitations include its dependency on parameter selection. A hybrid SVM presented in<sup>19</sup> for CSS combines ML with Red Deer Algorithm to optimize the spectrum sensing performance. The presented approach clusters secondary users then employ machine learning models for cluster head selection and spectrum classification. The optimization model is used to fine-tune the hyperparameters of SVM to enhance detection accuracy with low computational complexity. Simulation analysis under different SNR conditions exhibits the model better detection probability and error probability over traditional dual threshold and fuzzy energy detection approaches. However, the presented model exhibited sensitivity to initialization and scalability issues due to its centralized clustering and the computational demands of the optimization process.

In<sup>20</sup>, cooperative spectrum sensing is tackled with a scheme that applies the Adam optimiser to fuse several signal descriptors—Renyi entropy, signal energy, eigenvalue-based statistics, and Generalised Likelihood Ratio Test features—so that spectrum vacancies in cognitive radio networks can be identified more reliably. Experiments report markedly higher detection probabilities and lower false-alarm rates than those obtained by standard deep-neural-network approaches, although the method's success still hinges on careful initial tuning of Adam's parameters. Meanwhile<sup>21</sup>, proposes a multi-agent deep deterministic policy gradient (MADDPG) strategy for spectrum sensing. During a centralised training phase, secondary users exchange channel-state information; later, they choose channels independently, gravitating toward those with the greatest likelihood of being idle. This arrangement improves sensing accuracy and cuts collision rates relative to Q-learning and DQN baselines, while simultaneously curbing synchronisation and communication overhead. The chief limitation, however, is the substantial hyper-parameter tuning effort required when the framework is scaled to large networks.

The CSS approach presented in<sup>22</sup> for CRN utilizes CNN to classify different types of modulated signals from secondary users to define the spectrum usage. The generated and processed signals are used to train the CNN and classify them as signal or noise. The experimental analysis exhibits the enhanced spectrum sensing accuracy with minimum channel interference under different SNR conditions. However, the presented approach introduces additional overhead and sensing delays due to cooperative coordination and processing. The CSS prediction model presented in<sup>23</sup> integrates a LSTM model for local spectrum prediction in CRN. The presented model addresses the limitations like time-series dependency and sequence issues in traditional ML models. The parallel fusion-based cooperative spectrum prediction model with adaptive fusion rule is used to dynamically adjust to network conditions. This reduces the prediction errors caused by environmental factors and attains higher prediction accuracy over traditional approaches. However, the computational complexity of LSTM training and the integration of dynamic fusion parameters brings challenges for real-time scalability.

An improved CNN is combined with LSTM in<sup>24</sup> for spectrum sensing in CRN. This hybrid approach optimizes the number of neurons in DNN using simulated annealing to ensure efficient sensing under low SNR conditions. The CNN extracts spatial features and LSTM extracts the temporal dependencies to provide a robust solution in the process of identifying primary user activities. Experimental analysis exhibited the superior detection probabilities of the presented model compared to traditional ML models like SVM and kNN. However increased computation complexity and limited scalability are the limitations of the presented sensing model. The CSS model presented in<sup>25</sup> incorporates ResNet50 with a clustering approach to detect the spectrum holes with better detection accuracy. The presented model dynamically optimizes the spectrum utilization by identifying optimal unused bands by experimenting with different modulation schemes like BPSK, QPSK and 64-QAM with different SNR ranges. The experimental analysis highlights that the presented model with QAM provides better balance between the computational efficiency and accuracy at higher SNRs. However, the model failed to handle extreme congestion conditions which leads to reduced accuracy.

The hybrid deep learning model presented in<sup>26</sup> combines ResNet-LSTM for spectrum sensing in CRN. The strengths of ResNet are used for spatial feature extraction and LSTM is used for temporal feature learning. Using RadioML2016.10b with diverse signal conditions and modulation schemes the experimental analysis demonstrates better detection performance over traditional methods. However, the presented model requires higher training time and its computational complexity challenges the real-time implementation. The deep learning-based spectrum sensing model presented in<sup>27</sup> utilizes one dimensional CNN with BiLSTM to analyze the time series spectrum data. The presented model extracts the local features through one dimensional CNN and captures the temporal dependencies using BiLSTM. The classification results exhibit the presented model better performance over traditional approaches under low SNR conditions. Apart from the better performance, the model has limitations due to its sequential processing approach which increases the computational latency. Additionally, the BiLSTM's depth makes it less suitable for real-time or resource-constrained environments.

The hybrid deep learning approach presented in<sup>28</sup> incorporates CNN and RNN as an ensemble approach for spectrum sensing in CRN. The methodology incorporates CNN for spatial feature extraction and RNN for

temporal feature learning. This feature handling procedure aims to detect the primary user activities under challenging low SNR conditions. The system utilizes covariance matrices of signals as inputs and allows the model to learn energy and correlation features. The experimental analysis demonstrates the model better accuracy and reliability over techniques like energy detection and CNN-based methods specifically for the scenarios which have varying SNR and multiple secondary users. However, the presented model is dependency on covariance matrices and dense connectivity which increases the computational complexity and limits its deployment in dynamic or large-scale CRNs.

A deep reinforcement learning (DRL) based spectrum sensing CRN presented in<sup>29</sup> incorporates DRL with a Decentralized Partially Observable Markov Decision Process (Dec-POMDP). The presented approach effectively handles the noise uncertainty and provides enhanced detection performance. The spectrum holes are detected effectively utilizing the features that are automatically learned from dynamic environments. Experimental evaluation demonstrates that the presented model outperforms traditional spectrum sensing methods like deep Q-learning and CNN in terms of enhanced sensing accuracy and efficiency. However, the presented model has limitations as high computational demands in real-time applications due to their complexity. The CSS approach for CRN presented in<sup>30</sup> presents a Multi Featured Combination Network (MCN) by integrating CNN with GRU. The presented model utilizes CNN for spatial feature extraction and GRU for temporal feature modeling. The feature processing modules are arranged in a parallel structure to attain enhanced feature representation and fusion. Also to overcome the limitations of traditional binary decision-based fusion methods the presented model includes a Combination-Net that integrates node-specific confidence information to provide accurate decisions. The experimental results exhibit the model better detection accuracy under dynamic and low SNR conditions over conventional CSS schemes. The use of diverse modulation types for training ensures the model generalization ability across various signal environments.

A CNN-Transformer model is presented in<sup>31</sup> for cooperative spectrum sensing, combining convolutional feature extraction with Transformer-based temporal modeling. A posteriori probability ratio detector was designed under the Neyman–Pearson criterion to achieve optimal binary hypothesis testing. Experiments performed under Rician fading channels revealed detection probability improvements from 0.88 to 0.99 and about 6 dB gain compared with conventional energy detectors. However, the complexity of training the presented model and the dependence on centralized fusion increase the implementation challenges. A cooperative spectrum sensing model is presented in<sup>32</sup> using a double reconfigurable intelligent surface configuration. By optimizing the phase shifts of two RIS panels, the authors addressed non-convex optimization problems through successive convex approximation and a penalty-based convex–concave procedure. Simulation analysis confirmed significant improvements in cooperative detection when compared with baseline methods. Nevertheless, the approach adds considerable design complexity, requires precise placement of RIS units, and heavily relies on accurate channel state information for maintaining sensing reliability. A weighted graph-based approach is presented in<sup>33</sup> for cooperative spectrum sensing. A signal-to-weighted-graph mechanism was proposed to integrate multi-user data into a unified weighted graph, where graph sparsity was applied to characterize connectivity and derive a test statistic. Experimental results validated its advantage in low SNR environments relative to unweighted graph schemes. Despite its gains, the model's success depends on accurate weight estimation and threshold calibration, which may limit adaptability under highly variable channel conditions.

### Research gaps

The detailed research analysis presented above highlights the challenges in existing spectrum sensing procedures in CRNs. Existing spectrum sensing methods frequently face issues like shadowing, fading, noise uncertainty, and interference which affects the detection accuracy. While cooperative spectrum sensing approaches ease these challenges by utilizing spatial diversity. Traditional spectrum sensing algorithms, including energy detection, matched filtering, and cyclo-stationary feature detection depend on prior knowledge of signal characteristics which makes them unsuitable for dynamic network conditions. Deep learning (DL) models automatically learn signal features in the sensing process. But DL models exhibit limitations in effectively capturing the temporal dependencies and global correlations in time-series data. Conventional models like CNN variants are widely applied but these methods fail to integrate both spatial and temporal patterns especially in low SNR conditions. The hybrid architectures show significant improvements but face scalability issues and computational overhead which makes the model difficult to apply in a real-time environment. The existing procedures lack fluctuating environmental conditions and diverse signal modulation schemes. Furthermore, existing procedures fail to optimize the balance between detection probability and false alarm rates across varying sample lengths and modulation schemes. These research gaps highlight the need for a scalable, robust, and computationally efficient approach that not only captures spatial and temporal correlations but also adapts dynamically to varying channel and environmental conditions.

### Proposed work

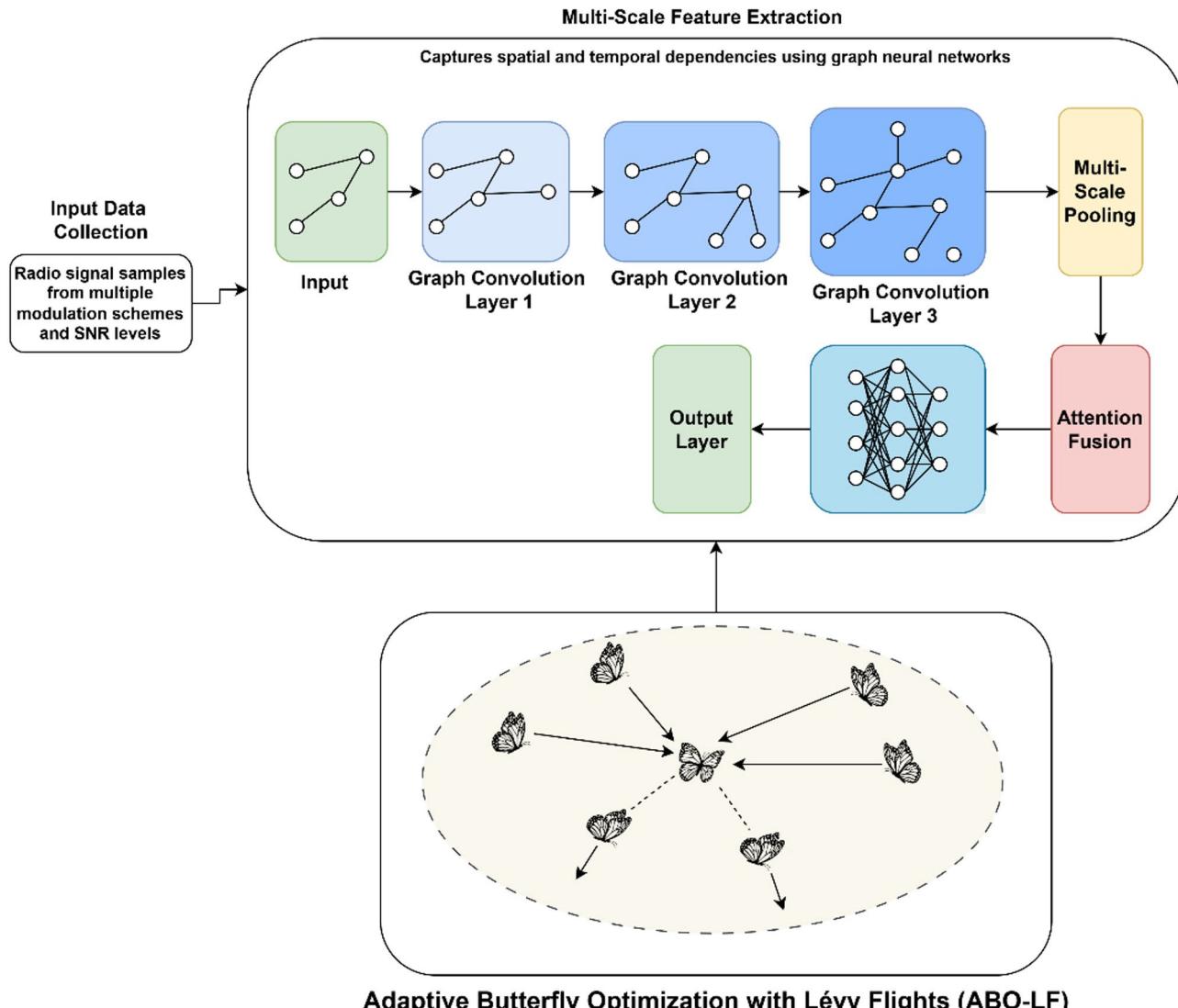
The proposed spectrum sensing model includes an Optimized Multi-Scale Graph Neural Network with Attention Mechanism (OMSGNNA) to efficiently extract and process spectral features. The process begins with signal inputs where each signal is converted into graph representations based on their spatial and spectral characteristics. The graph construction phase ensures that nodes represent key signal components and the edges capture relationships between signal features. The proposed architecture is composed of multiple graph convolutional layers to capture complex dependencies among nodes. Also, these layers employ an attention mechanism to arrange essential features and focus on relevant node connections. This enhances the model ability to differentiate the noise and useful signals in the sensing process.

To accommodate differences in feature scale and resolution, the network embeds multi-scale pooling layers that fuse representations drawn from several graph granularities, yielding features that remain expressive even

when signal conditions fluctuate. The pooled descriptor is subsequently routed into a dense (fully connected) layer that carries out the final detection task. Figure 1 sketches the end-to-end pipeline, comprising (i) signal pre-processing, (ii) graph construction, (iii) two attention-enhanced GNN stages, (iv) the multi-scale pooling block, and (v) the output classifier. Development of the Optimized Multi-Scale Graph Neural Network with Attention (OMSGNNA) therefore starts by clearly formulating the cooperative spectrum-sensing objective and encoding the raw observations in a graph-compatible format. In cognitive-radio settings, this task translates to identifying the modulation category of received signals across a range of Signal-to-Noise Ratios (SNRs). Formally, consider the sequence of in-phase ( $I$ ) and quadrature ( $Q$ ) components captured over  $T$  time steps is represented as

$$X = \{(I_t, Q_t)\}_{t=1}^T \quad (1)$$

where  $I_t \in R$  indicates the in-phase sample at time  $t$ ,  $Q_t \in R$  indicates the quadrature sample at time  $t$ ,  $T$  indicates the total number of time intervals. The goal is to classify each sample in  $X$  into a modulation type  $y \in C$ , where  $C$  is the set of modulation classes (e.g., AM, FM, QAM, PSK). The task can be expressed as finding a mapping  $f : X \rightarrow C$  in which  $f$  indicates a classifier learned from the dataset. To enhance feature



**Fig. 1.** Process flow of proposed model.

extraction by exploiting relationships between signal samples the data is represented as a graph. Define a graph  $G = (V, E, A)$  where  $V = \{v_1, v_2, \dots, v_T\}$  indicates set of  $T$  nodes each representing a sensing interval.  $E \subseteq V \times V$  indicates the edge set which defines the relationships between nodes.  $A \in R^{T \times T}$  indicates the adjacency matrix which captures the edge weights. Each node  $v_i$  is associated with a feature vector  $h_i^0 \in R^{d_0}$ , where  $d_0$  is the feature dimension is. The node features are constructed by combining the normalized I/Q samples which are mathematically formulated as follows

$$h_i^0 = [\widehat{I}_i, \widehat{Q}_i] \quad (2)$$

where  $\widehat{I}_i = \frac{I_i - \mu_I}{\sigma_I}$  indicates the standardized in-phase sample,  $\widehat{Q}_i = \frac{Q_i - \mu_Q}{\sigma_Q}$  indicates the standardized quadrature sample,  $\mu_I, \sigma_I$  indicates the mean and standard deviation of  $I$ ,  $\mu_Q, \sigma_Q$  indicates the mean and standard deviation of  $Q$ . The node feature matrix for all nodes is formulated as

$$H^0 = [h_1^0; h_2^0; \dots; h_T^0] \in R^{T \times d_0} \quad (3)$$

The edge weights are computed based on the correlation between features of nodes. The adjacency matrix  $A$  is defined as

$$A_{ij} = \begin{cases} \text{correlation}(h_i^0, h_j^0) & \text{if } h_i^0 \text{ and } h_j^0 \text{ are connected} \\ 0 & \text{Otherwise} \end{cases} \quad (4)$$

where *correlation* indicates the Pearson correlation coefficient which is formulated as

$$\text{correlation}(h_i^0, h_j^0) = \frac{\sum_{k=1}^{d_0} (h_{ik}^0 - \bar{h}_i^0)(h_{jk}^0 - \bar{h}_j^0)}{\sqrt{\sum_{k=1}^{d_0} (h_{ik}^0 - \bar{h}_i^0)^2} \sqrt{\sum_{k=1}^{d_0} (h_{jk}^0 - \bar{h}_j^0)^2}} \quad (5)$$

where  $\bar{h}_i^0$  indicates the mean of features for node  $i$ .  $k$  indicates the feature index. If the correlation is below a certain threshold, the edge is removed to promote sparsity. The resulting graph  $G = (V, E, A)$  is defined by  $H^0$  and it indicates the node feature matrix which contains the standardized I/Q samples.  $A$  indicates the adjacency matrix which captures the relationships between nodes.

### Graph neural network architecture

The Graph Neural Network (GNN) architecture in the proposed model extracts spatial relationships and hierarchical features from the graph representation of the input data. The Graph Convolutional Network (GCN) module serves as the backbone for processing graph-structured data in the proposed model. It operates by aggregating information from neighboring nodes while preserving graph connectivity, enabling the model to learn spatially-aware representations.

The input to a GCN layer is the graph  $G = (V, E, A)$  and a node feature matrix  $H^l \in R^{T \times d_l}$ , where  $V = \{v_1, v_2, \dots, v_T\}$  indicates the nodes of the graph with  $T$  nodes.  $E$  indicates the edges indicating connections between nodes.  $A \in R^{T \times T}$  indicates the adjacency matrix where  $A_{ij} > 0$  implies a connection between nodes  $i$  and  $j$ .  $H^l = [h_1^l, h_2^l, \dots, h_T^l]$  indicates the node feature matrix at layer  $l$ , where  $h_i^l \in R^{d_l}$

is the feature vector of node  $i$ . The goal of the GCN layer is to transform  $H^l$  into an updated feature matrix  $H^{l+1}$  by incorporating information from the graph structure encoded in  $A$ .

A key step in the GCN operation is to normalize the adjacency matrix to ensure consistent aggregation of neighbor information. The normalized adjacency matrix with self-loops is formulated as

$$\tilde{A} = D^{-1/2} (A + I) D^{-1/2} \quad (6)$$

where  $I \in R^{T \times T}$  indicates the identity matrix which adds self-loops to the graph.  $D \in R^{T \times T}$  indicates the diagonal degree matrix of  $(A + I)$  with  $D_{ii} = \sum_j (A_{ij} + \delta_{ij})$ .  $\delta_{ij}$  indicates the Kronecker delta which is equal to 1 if  $i = j$ , and 0 otherwise. The normalization ensures that the contribution of each neighbor is weighted inversely proportional to its degree, stabilizing the learning process.

The main operation in a GCN layer is the graph convolution which updates the feature matrix  $H^l$  using the graph structure. The update rule in the GCN layer is formulated as

$$H^{l+1} = \sigma(\tilde{A} H^l W^l) \quad (7)$$

where  $H^{l+1} \in R^{T \times d_{l+1}}$  indicates the updated node feature matrix at layer  $l + 1$ .  $\tilde{A} \in R^{T \times T}$  indicates the normalized adjacency matrix which encodes the graph connectivity.  $H^l \in R^{T \times d_l}$  indicates the node feature matrix at layer  $l$ .  $W^l \in R^{d_l \times d_{l+1}}$  indicates the learnable weight matrix which maps  $d_l$  dimensional input features to  $d_{l+1}$ dimensional output features.  $\sigma(\cdot)$  indicates the non-linear activation function which is applied

element-wise to introduce non-linearity into the representation. Further in the feature transformation process the input features  $H^l$  are linearly transformed using  $W^l$  which is mathematically expressed as

$$H_{trans} = H^l W^l \quad (8)$$

where each feature vector  $h_i^l$  is mapped into a new space of dimension  $d_{l+1}$ . The transformed features are aggregated using the normalized adjacency matrix  $A$  which is expressed as

$$H_{aggre} = \tilde{A} H_{trans} \quad (9)$$

where each node  $i$  aggregates the feature information from its neighbors weighted by the normalized connectivity. A non-linear activation function  $\sigma(\cdot)$  is applied to the aggregated features to introduce complexity into the model. Mathematically the process is expressed as

$$H^{l+1} = \sigma(H_{aggre}) \quad (10)$$

The output of the GCN layer is the updated feature matrix  $H^{l+1}$  where  $H^{l+1} = [h_1^{l+1}, h_2^{l+1}, \dots, h_T^{l+1}]$

and each  $h_i^{l+1} \in R^{d_{l+1}}$  contains the updated representation of node  $i$  enriched with information from its neighbors. To obtain deeper relationships in the graph, multiple GCN layers can be stacked. For a network with  $L$  layers the final output is

$$H^{final} = H^L \quad (11)$$

Each layer progressively refines the node representations by aggregating information over larger neighborhoods.

#### Multi-scale graph pooling

The Multi-Scale Graph Pooling operation in the proposed work is designed to reduce the size of the graph while retaining essential information about the graph's structure and node features. By pooling nodes in a hierarchical manner, the model captures local and global patterns which are crucial for tasks like classification and feature extraction in graph neural networks. Graph pooling reduces the number of nodes from  $T$  to  $T'$  ( $T' \leq T$ ) while preserving the graph's structural integrity and relevant feature information. The pooled representation allows the network to focus on significant nodes or regions which ensures efficient computation and better generalization.

Given an input graph  $G = (V, E, A)$  and node feature matrix  $H \in R^{T \times d}$  the pooling operation outputs. A reduced graph  $G' = (V', E', A')$  where  $V' \subseteq V$  and  $|V'| = T'$  indicates a reduced node feature matrix  $H_{po} \in R^{T' \times d}$ . The first step in graph pooling is assigning an importance score  $s_i$  to each node  $v_i \in V$ . This score determines the significance of each node for the downstream task. The importance score is computed as follows

$$s_i = q^\top \tanh(W_{pool} h_i + b) \quad (12)$$

where  $s_i \in R$  indicates the importance of score for node  $i$ .  $h_i \in R^d$  indicates the feature vector of node  $i$ .  $q \in R^d$  indicates the learnable query vector that emphasizes specific features.  $W_{po} \in R^{d \times d}$  indicates the weight matrix for the transformation.  $b \in R^d$  indicates the bias term.  $\tanh(\cdot)$  indicates the non-linear activation function which is used to ensure bounded values.

To ensure comparability across nodes the importance scores are normalized using a SoftMax function which is formulated as

$$\alpha_i = \frac{\expexp(s_i)}{\sum_{j=1}^T \expexp(s_j)} \quad (13)$$

where  $\alpha_i \in [0,1]$  indicates the normalized importance weight for node  $i$ .  $\expexp(\cdot)$  indicates the exponential function ensuring non-negativity. The weights  $\alpha_i$  represent the relative importance of each node in the graph. Nodes with the highest importance scores are selected to form the reduced graph  $G'$ . Let  $k$  represent the fraction of nodes to retain such that  $T' = \lfloor kT \rfloor$ . The retained nodes  $V' \subseteq V$  are  $V' = \{v_i \mid \text{top } T' \text{ nodes by } \alpha_i\}$ . This step ensures that the most relevant nodes determined based on the importance scores are preserved for further processing. The features of the retained nodes are aggregated to form the new feature matrix  $H_{po}$ . For each retained node  $v_i \in V'$  its updated feature vector is formulated as follows

$$h_i^{pool} = \sum_{j \in N(i)} \alpha_j h_j \quad (14)$$

where  $N(i)$  indicates the set of neighbors for node  $i$ .  $\alpha_j$  indicates the normalized importance score of neighbors  $j$ .  $h_j \in R^d$  indicates the feature vector of neighbor  $j$ . This weighted aggregation captures local context by combining features from neighboring nodes.

Further the adjacency matrix  $A$  is also reduced to reflect the connectivity of the retained nodes. The new adjacency matrix  $A' \in R^{T' \times T'}$  is derived by retaining rows and columns corresponding to the selected nodes which are formulated as

$$A' = A_{V', V'} \quad (15)$$

where  $A_{V', V'}$  is the submatrix of  $A$  corresponding to the indices of nodes in  $V'$ . To capture information at multiple scales pooling is applied at different layers of the GCN. Each pooling layer operates on the output of its preceding GCN layer producing a progressively coarser representation. For layer  $l$  the pooled output is formulated as

$$H_{pool}^l = \text{pool} (H^l, A^l) \quad (16)$$

where  $H^l \in R^{T_l \times d_l}$  indicates the node features at layer  $l$ .  $A^l \in R^{T_l \times T_l}$  indicates the adjacency matrix at layer  $l$ .

### Attention mechanism

The attention mechanism in the proposed architecture enhances the model's ability to focus on the most informative nodes and their relationships within the graph. By assigning dynamic weights to nodes and edges the attention mechanism captures both local importance and global dependencies. Node-level attention assigns weights to the edges between nodes in the graph, emphasizing the most relevant neighbors for feature aggregation. For a node  $v_i$  and its neighbor  $v_j$  the attention score  $\alpha_{ij}$  is computed as follows

$$e_{ij} = \text{LeakyReLU} (a^\top [Wh_i | Wh_j]) \quad (17)$$

where  $e_{ij} \in R$  indicates the raw attention score for the edge between nodes  $i$  and  $j$ .  $h_i, h_j \in R^d$  indicates the feature vectors of nodes  $i$  and  $j$  respectively.  $W \in R^{d \times d_{attn}}$  indicates the transformation matrix projecting the node features into an attention space of dimension  $d_{attn}$ .  $(|)$  indicates the concatenation operator which combines the transformed feature vectors of  $i$  and  $j$ .  $a \in R^{2d_{attn}}$  indicates the learnable vector that determines the importance of the concatenated features.  $\text{LeakyReLU}(\cdot)$  indicates the activation function which introduces non-linearity with a small slope for negative inputs. This score encodes the relevance of node  $j$  to node  $i$  based on their features and relationship. The raw attention scores  $e_{ij}$  are normalized using a SoftMax function to ensure they sum to 1 for each node is formulated as

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in N(i)} \exp(e_{ik})} \quad (18)$$

where  $\alpha_{ij} \in [0,1]$  indicates the normalized attention weight for the edge between  $i$  and  $j$ .  $N(i)$  indicates the set of neighbors of node  $i$ .  $\exp(\cdot)$  indicates the exponential function ensuring non-negativity. The normalized weight  $\alpha_{ij}$  represents the relative importance of node  $j$  in the feature aggregation process for node  $i$ . Further in the feature aggregation process, the updated feature vector for node  $i$  is computed as a weighted sum of its neighbors' features as follows

$$h'_i = \sigma \left( \sum_{j \in N(i)} \alpha_{ij} h_j \right) \quad (19)$$

where  $h'_i \in R^d$  indicates the updated feature vector for node  $i$ .  $\sigma(\cdot)$  indicates the activation function ReLU which is applied element-wise to introduce non-linearity. This operation allows node  $i$  to incorporate information from its most relevant neighbors, with the contribution of each neighbor controlled by  $\alpha_{ij}$ .

Global attention aggregates information from all nodes in the graph to produce a single graph-level embedding. These embedding captures global patterns and dependencies across the graph. In the process of global attention score calculation each node  $v_i$  is assigned a global attention score  $\beta_i$  based on its feature vector which is formulated as

$$\beta_i = \frac{\exp(q^\top h_i)}{\sum_{j=1}^T \exp(q^\top h_j)} \quad (20)$$

where  $\beta_i \in [0,1]$  indicates the normalized global attention weight for node  $i$ .  $h_i \in R^d$  indicates the feature vector of node  $i$ .  $q \in R^d$  indicates the learnable query vector emphasizing specific features. The SoftMax normalization ensures that the global attention scores  $\beta_i$  sum to 1 across all nodes. The global graph embedding  $g$  is computed as a weighted sum of the node features as follows

$$g = \sum_{i=1}^T \beta_i h_i \quad (21)$$

where  $g \in R^d$  indicates the graph-level representation.  $T$  indicates the total number of nodes in the graph. These embedding aggregates information from all nodes, weighted by their global importance. To improve the expressive power of the attention mechanism, multi-head attention can be employed. In this approach, multiple attention heads are computed independently, and their outputs are concatenated

$$h_i^{\text{multi-head}} = \oplus_{m=1}^M h_i^{(m)} \quad (22)$$

where  $h_i^{multi-head} \in R^{M \cdot d}$  indicates the combined feature vector from  $M$  attention heads.  $h_i^{(m)}$  indicates the output of the  $m^{th}$  attention head for node  $i$ . Each attention head learns a different aspect of node relationships, enhancing the model's capacity to capture diverse patterns.

### Multi-scale feature fusion

Multi-Scale Feature Fusion is a critical step in the proposed model that combines information from multiple Graph Neural Network (GCN) layers. Each layer captures features at a different level of abstraction from localized information in the initial layers to more global patterns in the deeper layers. By integrating these features the model benefits from both fine-grained and coarse-grained information, improving its representational power. Graph Neural Networks learn hierarchical representations, with each layer aggregating information from increasingly larger neighborhoods in the graph. However, depending on a single layer's output may overlook valuable information captured at other layers. Multi-Scale Feature Fusion addresses this by combining outputs from all GCN layers. Mathematically it is formulated as

$$H_{fusion} = \bigoplus_{l=1}^L H^l \quad (23)$$

where  $H_{fusion} \in R^{T \times \sum_{l=1}^L d_l}$  indicates the final fused feature matrix combining outputs from all  $L$  layers.

$H^l \in R^{T \times d_l}$  indicates the node feature matrix from the  $l^{th}$  layer.  $T$  indicates the number of nodes in the graph.  $d_l$  indicates the number of features in the  $l^{th}$  layer. This fusion ensures that the final representation retains both local and global graph information. The output of a GCN layer at depth  $l$  is computed as

$$H^l = \sigma \left( \tilde{A} H^{l-1} W^l \right) \quad (24)$$

where  $H^l \in R^{T \times d_l}$  indicates the node feature matrix after the  $l^{th}$  GCN layer.  $\tilde{A} \in R^{T \times T}$  indicates the normalized adjacency matrix with self-loops.  $H^{l-1} \in R^{T \times d_{l-1}}$  indicates the input node feature matrix from the  $(l-1)^{th}$  layer.  $W^l \in R^{d_{l-1} \times d_l}$  indicates the learnable weight matrix for the  $l^{th}$  layer.  $\sigma(\cdot)$  indicates the non-linear activation function. Each layer computes features that aggregate information from progressively larger neighborhoods in the graph leading to hierarchical representations. The features from all GCN layers are concatenated along the feature dimension. Let  $H^1, H^2, \dots, H^L$  represent the outputs of the  $L$  GCN layers. The concatenated feature matrix is formulated as

$$H_{fusion} = [H^1 | H^2 | \dots | H^L] \quad (25)$$

where  $(|)$  indicates the concatenation operator.  $H_{fusion} \in R^{T \times \sum_{l=1}^L d_l}$  indicates the combined feature matrix with dimensions reflecting the total features across all layers. Concatenation preserves the unique information captured at each layer, providing a richer representation. In cases where certain layers are more important, a weighted fusion approach can be used which is mathematically expressed as

$$H_{fusion} = \sum_{l=1}^L w_l H^l \quad (26)$$

where  $w_l \in R$  indicates the learnable weight assigned to the  $l^{th}$  layer.  $\sum_{l=1}^L w_l = 1$  which ensures the weights are normalized. This approach allows the model to emphasize features from layers that contribute more to the task. The fused feature matrix  $H_{fusion}$  is used as input to the classification or decision-making layers. For a classification task, a fully connected layer with softmax activation maps the fused features to class probabilities

$$Y = softmax(H_{fusion} W_f) \quad (27)$$

where  $Y \in R^{T \times C}$  indicates the output class probabilities for  $C$  classes.  $W_f \in R^{\sum_{l=1}^L d_l \times C}$  indicates the weight matrix of the fully connected layer. This step ensures that the multi-scale fused features contribute directly to the task, leveraging the rich hierarchical information captured across layers.

### Output of the GNN

The Graph Neural Network (GNN) architecture's final step is to produce the output by processing the fused features obtained from multi-scale feature fusion. This step maps the node to task-specific outputs such as class probabilities. The input to this stage is the fused feature matrix  $H_{fusion}$  derived from multi-scale feature fusion

$$H_{fusion} \in R^{T \times \sum_{l=1}^L d_l} \quad (28)$$

where  $T$  indicates the number of nodes in the graph.  $L$  indicates the total number of GCN layers.  $d_l$  indicates the feature dimension at layer  $l$  and  $\sum_{l=1}^L d_l$  is the combined feature dimension after fusion. Depending on the task, the output can be node-level predictions, graph-level predictions, or embeddings for downstream tasks. The fused feature matrix is processed through a fully connected layer to map it to the desired output space. For classification tasks, this mapping is achieved as per the formulation given below

$$Y = H_{fusion}W_f + b_f \quad (29)$$

where  $Y \in R^{T \times C}$  indicates the output matrix where each row corresponds to the logits for  $C$  classes for a node.  $W_f \in R^{\sum_{l=1}^L d_l \times C}$  indicates the learnable weight matrix of the fully connected layer.  $b_f \in R^C$  indicates the learnable bias vector.  $C$  indicates the number of classes for classification tasks. This linear transformation projects the high-dimensional fused features into the task-specific output dimension.

For classification tasks, the logits  $Y$  are converted into class probabilities using the SoftMax activation function which is formulated as

$$P_{i,c} = \frac{\expexp(Y_{i,c})}{\sum_{j=1}^C \expexp(Y_{i,j})} \quad (30)$$

where  $P_{i,c} \in [0,1]$  indicates the probability of node  $i$  belonging to class  $c$ .  $Y_{i,c}$  indicates the logit value for node  $i$  and class  $c$ .  $\sum_{j=1}^C P_{i,j} = 1$  ensures that the probabilities for each node sum to 1 across all classes. The SoftMax function normalizes the logits into a probability distribution for multi-class classification. The model's predictions are compared to the true labels using an appropriate loss function. For a multi-class classification task, the cross-entropy loss is commonly used

$$L = -\frac{1}{T} \sum_{i=1}^T \sum_{c=1}^C y_{i,c} \log(P_{i,c}) \quad (31)$$

where  $L$  indicates the average loss over all nodes.  $y_{i,c} \in \{0,1\}$  indicates the ground-truth label for node  $i$  and class  $c$  (1 if true, 0 otherwise).  $P_{i,c}$  indicates the predicted probability for node  $i$  belonging to class  $c$ . Minimizing this loss function adjusts the model parameters to improve classification accuracy.

For tasks requiring a single prediction for the entire graph (e.g., graph classification), a graph-level embedding  $g$  is first derived by aggregating node embeddings:

$$g = \sum_{i=1}^T \beta_i h_i^{fusion} \quad (32)$$

where  $\beta_i \in [0,1]$  is the weight assigned to node  $i$  computed using a global attention mechanism.  $h_i^{fso} \in R^{\sum_{l=1}^L d_l}$  indicates the fused feature vector for node  $i$ . The graph-level embedding  $g$  is then passed through fully connected layers to produce the desired output such as logits for graph classification

$$Y_{gra} = gW_{gra} + b_{gra} \quad (33)$$

where  $Y_{gra} \in R^C$  indicates the logits for the graph.  $W_{gra} \in R^{\sum_{l=1}^L d_l \times C}$  indicates the weight matrix for graph-level predictions.  $b_{gra} \in R^C$  indicates the bias vector. To prevent overfitting, regularization techniques dropout is applied. In the dropout process, nodes or features are dropped randomly during training which prevents the model from relying too heavily on specific nodes. Mathematically it is formulated as

$$H_{Dropout} = Dropout(H_{fusion}, p) \quad (34)$$

where  $p$  indicates the dropout rate. In order to attain improved performance, the proposed MGNNA model includes a hybrid optimization algorithm.

### Adaptive butterfly optimization with Lévy flights (ABO-LF)

Adaptive Butterfly Optimization with Lévy Flights (ABO-LF) is an advanced metaheuristic optimization algorithm inspired by the foraging behavior of butterflies. It combines the principles of adaptive attraction and Lévy flight dynamics to balance exploration and exploitation in the hyperparameter search process. The algorithm begins by initializing a population of butterflies, each representing a potential solution in the hyperparameter search space. Let  $x_i^0 = \{\theta_1, \theta_2, \dots, \theta_n\}$ ,  $i = 1, 2, \dots, N$  where  $x_i^0 \in R^n$  indicates the position of the  $i^{th}$  butterfly in the  $n$  dimensional search space representing hyperparameters like learning rate, GCN depth.  $N$  indicates the total number of butterflies (population size).  $\theta_j$  indicates the value of the  $j^{th}$  hyperparameter. The fitness of each butterfly is evaluated using the objective function  $f(x_i)$  which measures the quality of the hyperparameters. Butterflies detect and move toward more promising solutions based on their "scent," which is influenced by both local and global stimuli. The scent strength  $S_i$  of the  $i^{th}$  butterfly is mathematically expressed as

$$S_i = I_i \cdot F^\beta \quad (35)$$

where  $S_i \in R$  indicates the scent strength of the  $i^{th}$  butterfly.  $I_i = f(x_i)$  indicates the stimulus intensity which is the fitness value of the butterfly.  $F \in R$  indicates the scaling factor that decreases over iterations balancing exploration and exploitation.  $\beta \in [0,1]$  is the control parameter determining the influence of the scaling factor. As iterations progress  $F$  decreases, reducing the range of movement and encouraging fine-tuning of solutions.

Each butterfly updates its position either globally or locally based on the attraction mechanism. When a butterfly is attracted to the best-known solution in the population:

$$x_i^{new} = x_i^{current} + S_i \cdot (x_{best} - x_i^{current}) \quad (36)$$

where  $x_i^{new}, x_i^{current} \in R^n$  indicates the new and current positions of the  $i^{th}$  butterfly.  $(x_{bs} \in R^n)$  indicates the position of the global best butterfly in the population.  $(S_i)$  indicates the scent strength influencing the movement magnitude.

When a butterfly is attracted to a neighboring butterfly

$$x_i^{new} = x_i^{current} + S_i \cdot (x_j - x_i^{current}) \quad (37)$$

where  $x_j$  indicates the position of a neighboring butterfly  $j$ , chosen randomly. Local attraction ensures diversity in the search process, preventing premature convergence. To enhance exploration and escape local optima, Lévy flight dynamics introduce stochastic jumps in the butterfly's movement. The updated position is

$$x_i^{new} = x_i^{current} + \alpha \cdot \text{Lévy}(\lambda) \quad (38)$$

where  $\alpha \in R$  indicates the step size which controls the magnitude of the jump.  $\text{Lévy}(\lambda)$  indicates the Lévy flight distribution characterized by long-tailed step sizes. It is defined as

$$\text{Lévy}(\lambda) \sim \frac{1}{|u|^{1+\lambda}} \quad (39)$$

where  $\lambda \in (0,2]$  is a scaling parameter. Lévy flight allows the butterfly to make large, random jumps, ensuring global search capabilities. The algorithm dynamically adjusts the balance between exploration (global search) and exploitation (local refinement) using an adaptive weighting scheme

$$w_{global} = e^{-\gamma t}, \quad w_{local} = 1 - w_{global} \quad (40)$$

where  $w_{global}, w_{local} \in [0,1]$  indicates the weights controlling the influence of global and local attraction.  $t$  indicates the current iteration.  $\gamma \in R^+$  indicates the decay rate determining the rate of transition from exploration to exploitation. Early iterations emphasize global search ( $w_{global}$  high), while later iterations favor local refinement ( $w_{local}$  high). The optimization process terminates when it reaches maximum iterations or the solutions converge. The best solution found during the optimization process is returned as the optimal set of hyperparameters. In the proposed OMSGNNA framework, this mathematical formulations play a essential role in connecting the design with experimental validation. The formulations for signal preprocessing (I/Q normalization and graph construction) clearly define how raw communication signals are transformed into graph-structured data. The graph convolution and pooling equations demonstrate how spatial and temporal dependencies are aggregated across nodes, ensuring robust feature extraction. Attention mechanism equations mathematically highlight how the model assigns importance to relevant nodes and edges, strengthening detection in low SNR conditions. The fusion and classification equations validate the mapping of multi-scale features into final decision probabilities, directly linking to detection accuracy metrics. Finally, the adaptive butterfly optimization equations formalize the hyperparameter tuning process, ensuring balanced exploration and convergence during training.

## Results and discussion

The proposed Optimized Multi-Scale Graph Neural Network with Attention Mechanism (OMSGNNA) was implemented and tested in Python. To ensure reproducibility, the configuration parameters used for evaluating the proposed OMSGNNA model are summarized in Table 1. The simulation was implemented in Python using PyTorch and PyTorch Geometric, with hyperparameters tuned by the Adaptive Butterfly Optimization with Lévy Flights (ABO-LF). The dataset employed was RadioML2016.10b, covering 11 modulation schemes with SNR levels ranging from  $-20$  dB to  $+18$  dB. The dataset was split into 80% for training and 20% for testing. Hyper-parameter tuning was guided by the Adaptive Butterfly Optimization algorithm enhanced with Lévy flights, which systematically searched for optimal values of the learning rate, the number of GCN layers, and the count of attention heads. Experiments were run over a range of SNR settings to mimic real-world noise and assess model resilience. Classification effectiveness was gauged through accuracy, precision, recall, and F1-score. To verify robustness, the trained network was benchmarked against traditional baseline architectures. A complete list of the simulation hyper-parameters employed in these experiments appears in Table 1.

The evaluation relied on the RadioML2016.10b benchmark dataset<sup>34</sup>, which supplies complex base-band waveforms expressed as in-phase (I) and quadrature (Q) sequences—ideal for modulation-recognition and cooperative spectrum-sensing studies. Each record is annotated with both its modulation category and a signal-to-noise ratio that varies from  $-20$  dB to  $18$  dB. The corpus spans eleven modulation families—AM-DSB, AM-SSB, FM, GFSK, CPFSK, BPSK, QPSK, 8-PSK, PAM-4, 16-QAM, and 64-QAM—and maintains an even sample distribution across SNR levels. During preprocessing, the raw I/Q vectors are recast as graph signals: nodes correspond to discrete sensing windows, while edges capture inter-feature relationships quantified via Pearson correlation. For assessment, 80% of the data was reserved for training and the remaining 20% for testing, as summarized in Table 2.

Model	Parameter	Range / type
Proposed MS-GNN-Attn	Learning rate	0.001
	Optimizer	RMSprop
	GCN layers	3
	Attention heads	4
	Pooling method	Multi-scale graph pooling
	Dropout rate	0.5
	Loss function	Adaptive margin loss
	Batch size	64
	Training epochs	100
	Hyperparameter tuning	Adaptive butterfly optimization with Lévy flights (ABO-LF)
GCN	Activation function	ReLU
	Learning rate	0.001
	Optimizer	Adam
	GCN layers	2
	Dropout rate	0.4
	Batch size	64
GAT	Activation function	ReLU
	Learning rate	0.001
	Optimizer	Adam
	GAT layers	2
	Attention heads	2
	Dropout rate	0.4
MLP	Batch size	64
	Activation function	LeakyReLU
	Learning rate	0.001
	Optimizer	Adam
	Hidden layers	2
	Neurons per layer	[128, 64]
MLP	Dropout rate	0.3
	Batch size	64
	Activation function	ReLU

**Table 1.** Simulation hyperparameters.

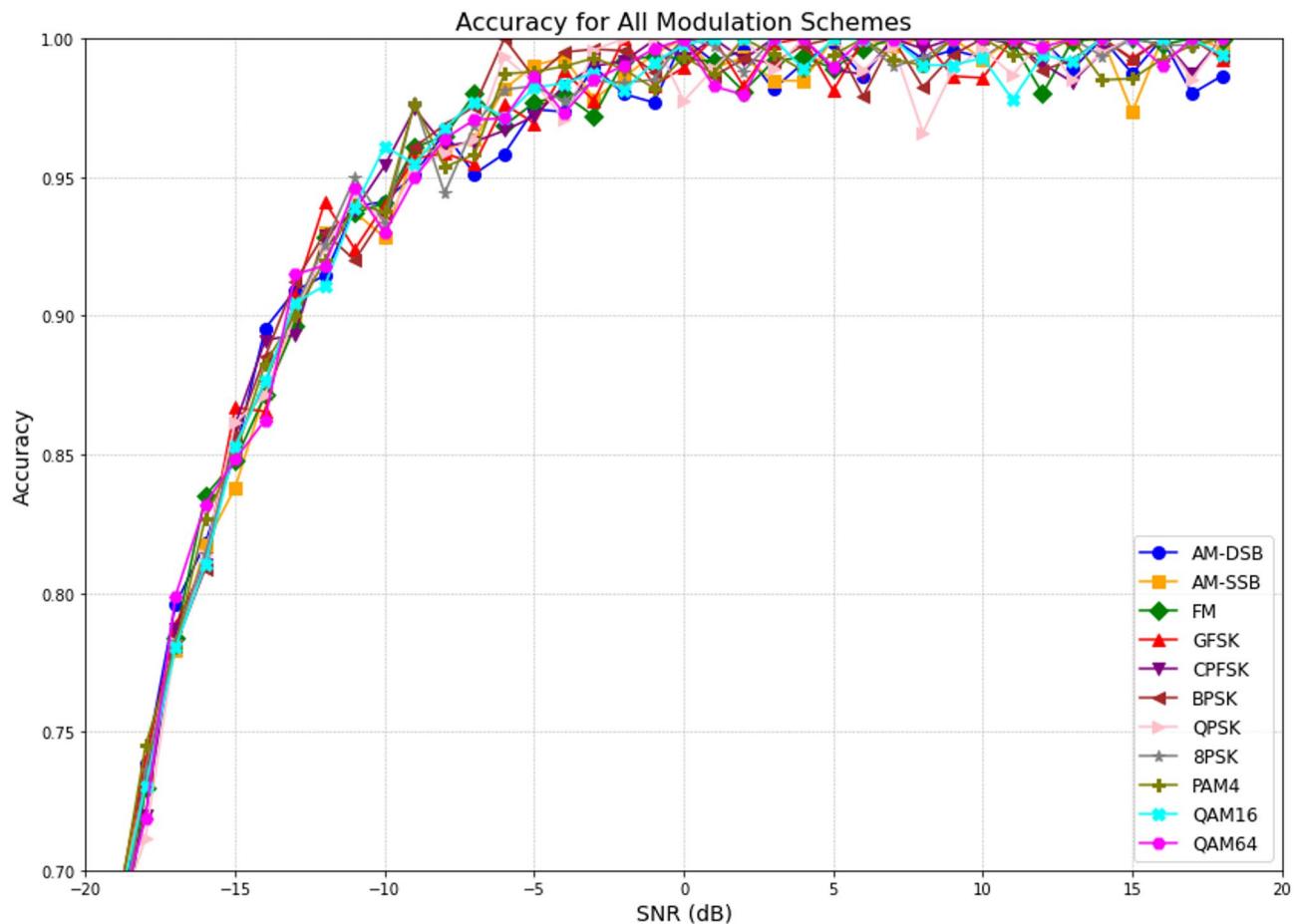
The detection accuracy analysis given in Fig. 2 for various modulation schemes across different Signal-to-Noise Ratio (SNR) demonstrates the proposed method's overall high accuracy in low-SNR environments. The result indicates the model's robustness to noise and interference. As SNR increases all the modulation schemes exhibit improved performance as higher SNR reduces the influence of noise on signal detection. Among the modulation schemes AM-DSB and AM-SSB exhibit high accuracy across the SNR range due to their simpler spectral properties which further simplifies the detection. Whereas PAM4 and QAM64 exhibit comparatively lower accuracy, especially at lower SNRs due to their complex structures and higher susceptibility to noise-induced symbol errors.

The precision analysis given in Fig. 3 highlights the ability of the proposed system to accurately identify true positive instances across various modulation schemes at different SNR levels in the spectrum sensing process. As SNR increases the precision of all modulation schemes improves significantly due to reduced noise interference. AM-DSB and FM exhibit superior precision across the SNR range as they are less prone to false positives. The modulation techniques like PAM4 and QAM64 exhibit lower precision particularly at low SNR levels due to increased misclassification rates. The advanced feature extraction and noise resilient classification techniques enhance the proposed model's ability to distinguish similar modulation patterns. Overall, the analysis highlights the proposed model robustness and adaptability in varying SNR conditions and modulation complexities.

The recall analysis given in Fig. 4 evaluates the proposed model performance of various modulation schemes in terms of correctly detecting true positives. As the SNR increases, recall values for all modulation schemes increase and it reflects the proposed model better detection capabilities under reduced noise conditions. Specifically, AM-DSB and FM exhibit higher recall across most SNR levels whereas complex schemes like QAM64 and PAM4 demonstrate lower recall, particularly at lower SNR values due to their complex signal representations which are susceptible to noise-induced misdetections. The proposed approach achieves consistent recall improvement due to the advanced signal processing and learning techniques. This allows the proposed model to attain enhanced classification performance by capturing relevant features even in noisy environments.

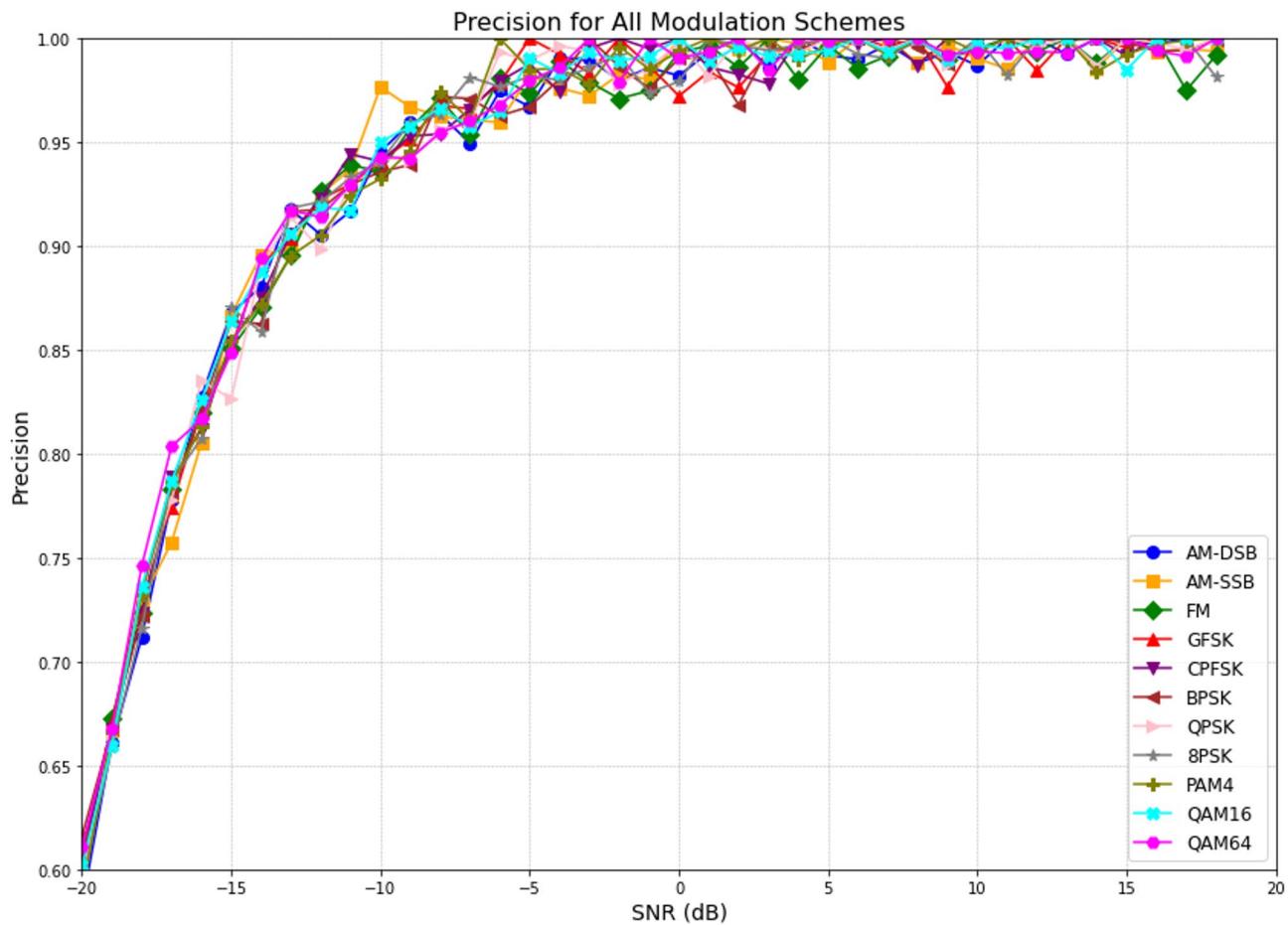
The F1-score analysis given in Fig. 5 evaluates the proposed model balance between precision and recall for various modulation schemes. The F1-score increases with higher SNR levels which reflects the improved

Modulation type	SNR (dB)	Training samples	Testing samples
AM-DSB	-20 to 18	12,800	3200
AM-SSB	-20 to 18	12,800	3200
FM	-20 to 18	12,800	3200
GFSK	-20 to 18	12,800	3200
CPFSK	-20 to 18	12,800	3200
BPSK	-20 to 18	12,800	3200
QPSK	-20 to 18	12,800	3200
8PSK	-20 to 18	12,800	3200
PAM4	-20 to 18	12,800	3200
QAM16	-20 to 18	12,800	3200
QAM64	-20 to 18	12,800	3200
Total	-20 to 18	140,800	35,200

**Table 2.** Dataset description.**Fig. 2.** Detection accuracy for different modulation schemes.

detection accuracy and reduced false positives, negatives of the proposed approach. AM-DSB and FM exhibit superior F1-scores across most SNR levels, whereas QAM64 and PAM4 exhibit lower F1-scores in low SNR conditions. The proposed method exhibits its significant advantage by maintaining the high F1-scores even in noisy environments. Also, the advanced feature selection and classification strategies of proposed model effectively balance precision and recall which ensures accurate spectrum sensing across diverse modulation schemes and SNR conditions.

Figure 6 depicts the missed detection rate analysis of the proposed model considering different modulation schemes in spectrum sensing. The missed detection rate is obtained by measuring the proportion of undetected signals under varying SNR conditions. A lower missed detection rate indicates the better performance of the



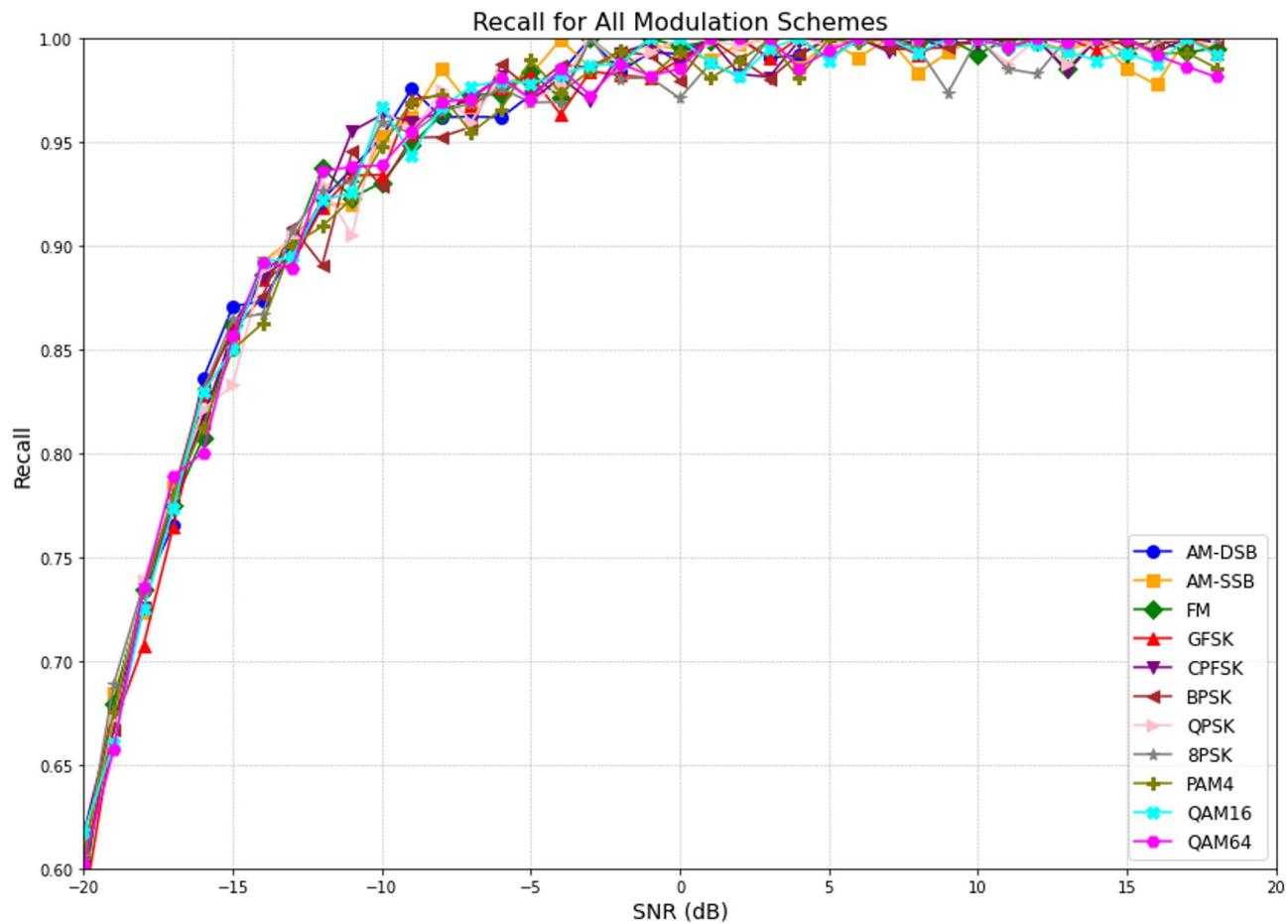
**Fig. 3.** Precision for different modulation schemes.

proposed model in identifying signals even in noisy environments. The results exhibit the significant reduction in missed detection rates as SNR increases which further reflects in the improved signal clarity and recognition accuracy. Modulations like AM-DSB and FM demonstrate superior performance with the lowest missed detection rates across all SNR levels due to their simpler waveform structures whereas QAM64 and PAM4 exhibit higher missed detection rates at lower SNRs.

Figure 7 contrasts the precision achieved by the proposed OMSGNNA with three baseline methods—Graph Convolutional Network (GCN), Graph Attention Network (GAT) and a Multilayer Perceptron (MLP)—across a range of SNR values. At the lowest SNR of  $-20$  dB, OMSGNNA secures a precision close to 0.55, exceeding the scores of GCN (0.5), GAT (0.53) and MLP (0.45). When the SNR rises to  $0$  dB, OMSGNNA precision climbs to 0.91, while GCN, GAT and MLP register 0.84, 0.85 and 0.79, respectively. In high-SNR settings, the proposed model approaches an optimal precision of 0.98, decisively outperforming GCN (0.88), GAT (0.88) and MLP (0.81). Across every SNR tested, OMSGNNA delivers the most accurate results, highlighting its robustness under noisy as well as favorable channel conditions.

The superior performance of OMSGNNA is highlighted due to its advanced graph structure and feature representation capabilities which enables it to capture dependencies and patterns in noisy environments. Whereas existing GCN and GAT exhibit moderate performance due to their limited adaptability to low SNRs due to less efficient feature aggregation. The least performance is exhibited by MLP as it lacks the ability to model graph relationships, which makes it less robust under low SNRs. The proposed OMSGNNA ability in achieving high precision even at low SNRs makes it highly effective for spectrum sensing tasks in noisy and dynamic wireless environments.

Figure 8 depicts the recall comparative analysis of the proposed OMSGNNA model against existing approaches under varying SNR conditions in the spectrum sensing process. At an SNR of  $-20$  dB, the proposed OMSGNNA achieves a recall of approximately 0.60 and outperforms GCN recall of 0.55, GAT recall of 0.58, and MLP recall of 0.50. As the SNR increases to  $0$  dB, OMSGNNA exhibits a recall of 0.92 which is higher than GCN recall of 0.85, GAT recall of 0.85, and MLP recall of 0.78. At high SNRs the recall of OMSGNNA stabilizes at 0.98, whereas GCN and GAT exhibits 0.88. The recall of MLP remains lower at 0.80 even at high SNR. The proposed OMSGNNA exhibits robustness and adaptability and maintains high recall across a wide range of SNRs. This indicates the highly reliable performance of the proposed approach for spectrum sensing in dynamic CRN environments.

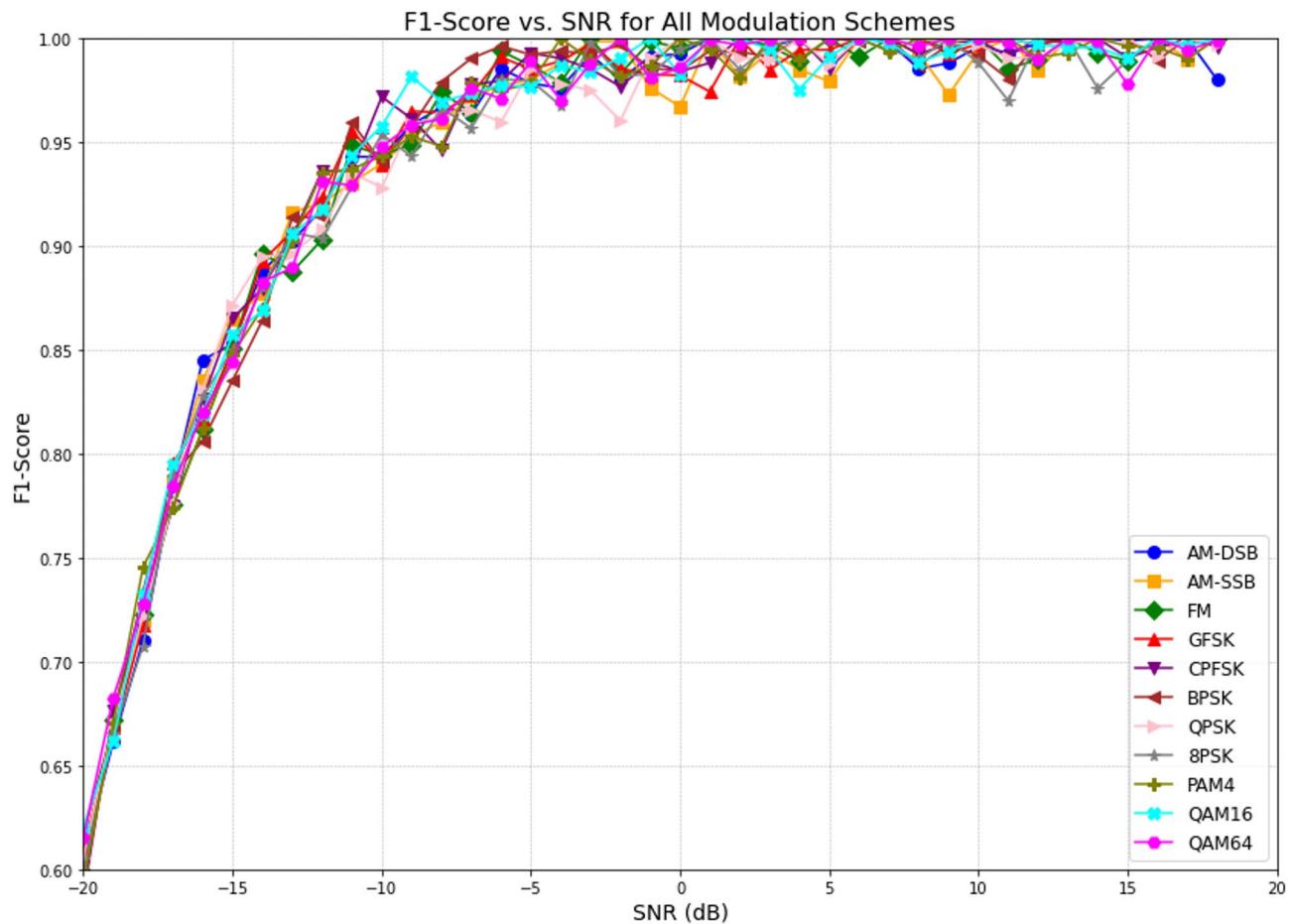


**Fig. 4.** Recall for different modulation schemes.

The comparative analysis given in Fig. 9 depicts the F1-Score performance of the proposed OMSGNNA model in the spectrum sensing process across varying SNR levels. At an SNR of -20 dB the proposed OMSGNNA achieves an F1-Score of approximately 0.57, surpassing GCN F1-score of 0.52, GAT F1-score of 0.55, and MLP F1-score of 0.48. As the SNR improves to 0 dB the F1-Score of proposed OMSGNNA increases significantly to 0.92, outperforming GCN F1-score of 0.84, GAT F1-score of 0.85, and MLP F1-score of 0.78. At high SNR levels such as 15 dB the proposed OMSGNNA model reaches an F1-Score of 0.98 whereas existing GCN and GAT exhibits 0.88 and and MLP exhibits 0.8 which is lesser than the proposed OMSGNNA.

The comparative analysis of missed detection rate (MDR) against SNR given in Fig. 10 highlights the superior performance of the proposed OMSGNNA model over existing techniques in the spectrum sensing process. At a low SNR of -20 dB the proposed OMSGNNA exhibits an MDR of 0.4 which is lower than GCN of 0.45, GAT of 0.42, and MLP of 0.5. As the SNR improves to 0 dB the proposed OMSGNNA model MDR further reduces to 0.08 and outperforms GCN and GAT attains 0.15, and MLP which attains 0.22. At high SNR levels the proposed OMSGNNA attains low MDR of 0.02, compared to GCN and GAT which attains 0.12, and MLP which attains 0.2. The proposed OMSGNNA superior performance is highlighted due to its enhanced ability in extracting and processing spectral features through optimized message passing and attention mechanisms. These advanced techniques enable the model to handle noisy and complex environments effectively and result in reduced MDR across all SNR levels.

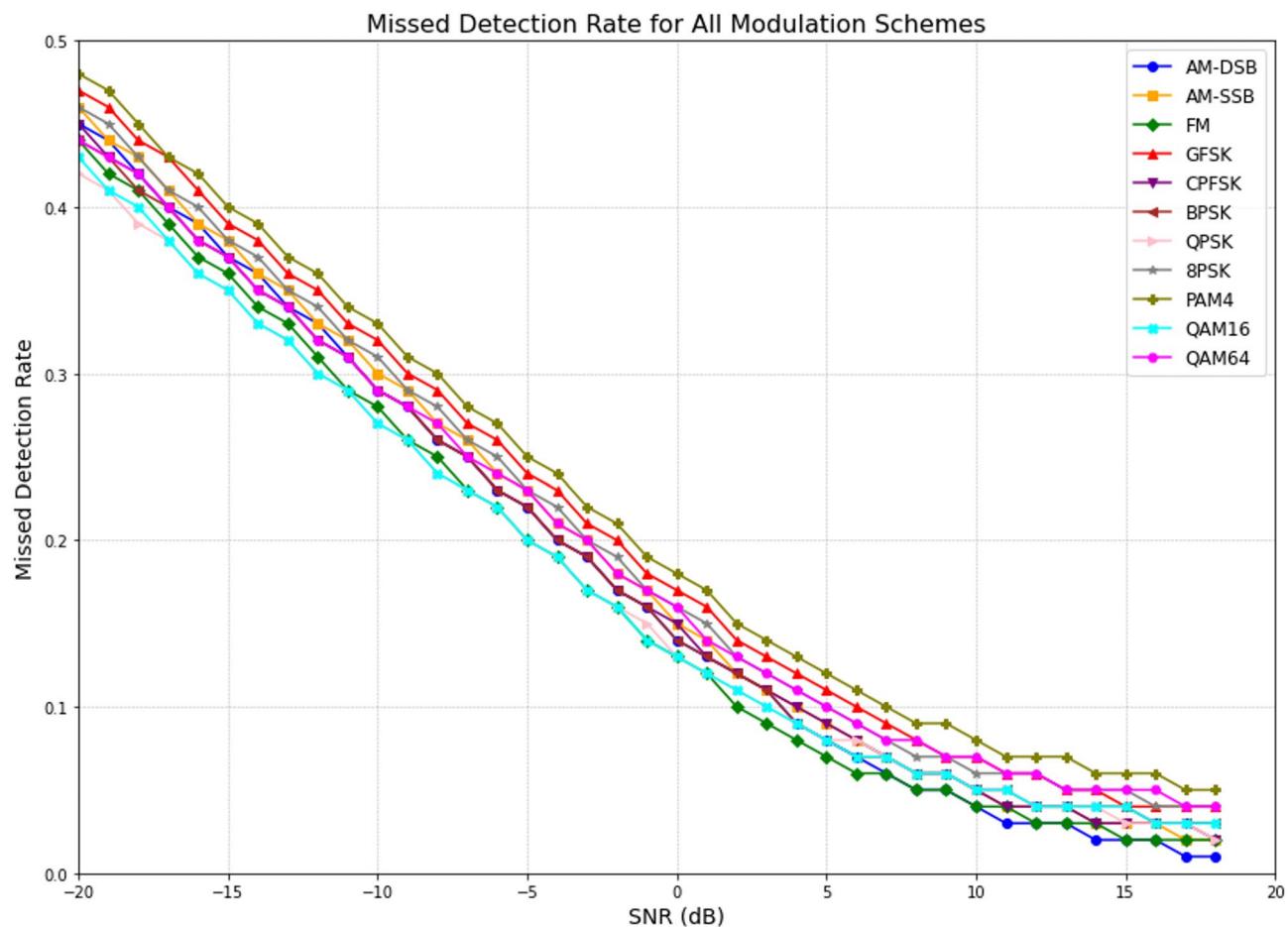
Figure 11 clearly illustrates that the proposed OMSGNNA outshines the benchmark models across all signal-to-noise ratios. When the SNR is as low as -20 dB, OMSGNNA secures 0.58 accuracy, which is better than GCN (0.55), GAT (0.57), and MLP (0.50). At a moderate SNR of 0 dB, its accuracy climbs to 0.93, surpassing the 0.89, 0.87, and 0.79 achieved by GCN, GAT, and MLP, respectively. Under high-SNR conditions, OMSGNNA peaks at 0.98 accuracy, outperforming GCN (0.96), GAT (0.93), and MLP (0.83). These gains stem from OMSGNNA's multi-scale graph architecture, which captures intricate spectral relationships even in noisy environments, while attention mechanisms and the Adaptive Butterfly Optimization with Lévy Flights further bolster resilience. Collectively, the findings affirm OMSGNNA's capability to deliver highly reliable spectrum sensing across varied SNR levels, positioning it as a strong candidate for deployment in cognitive radio networks.



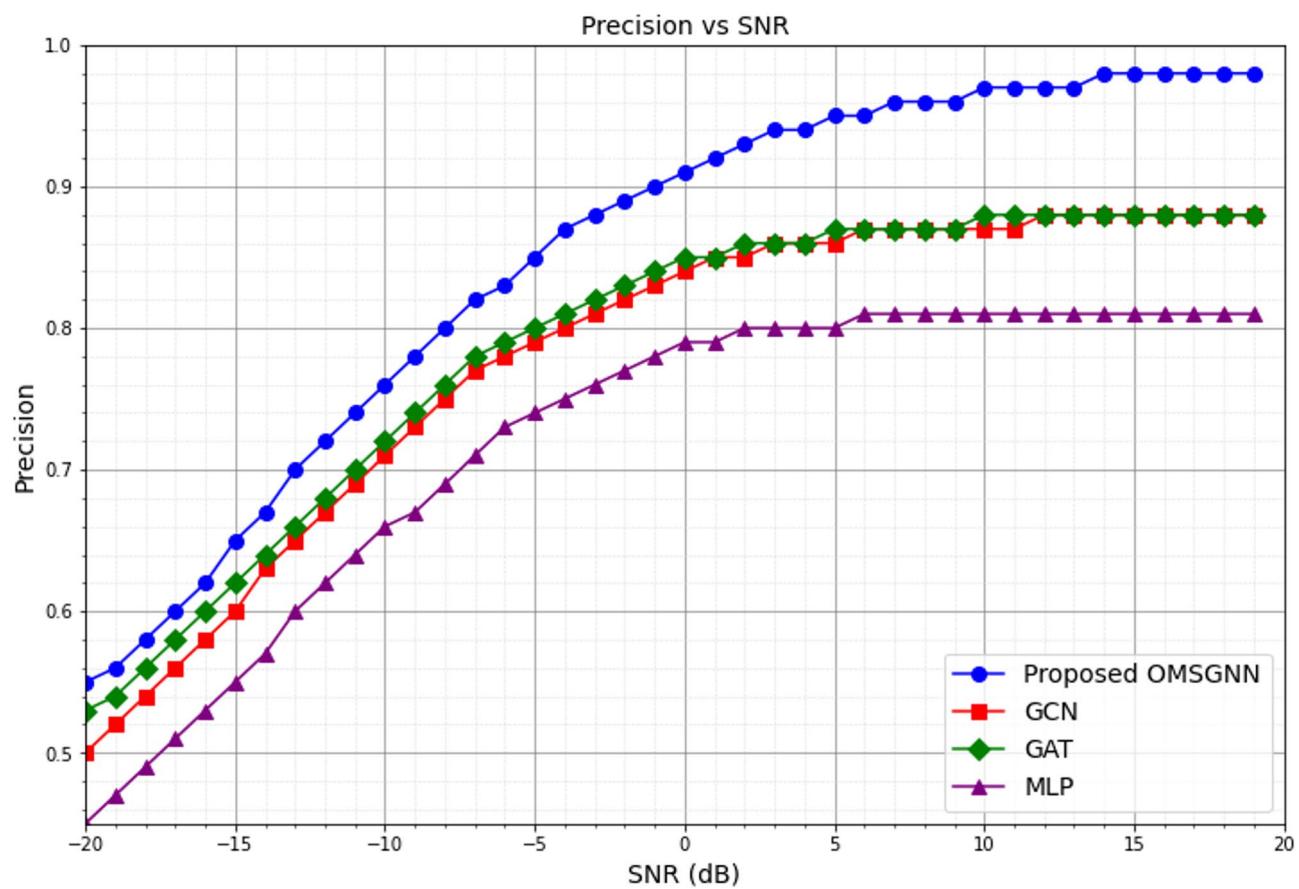
**Fig. 5.** F1-Score for different modulation schemes.

## Conclusion

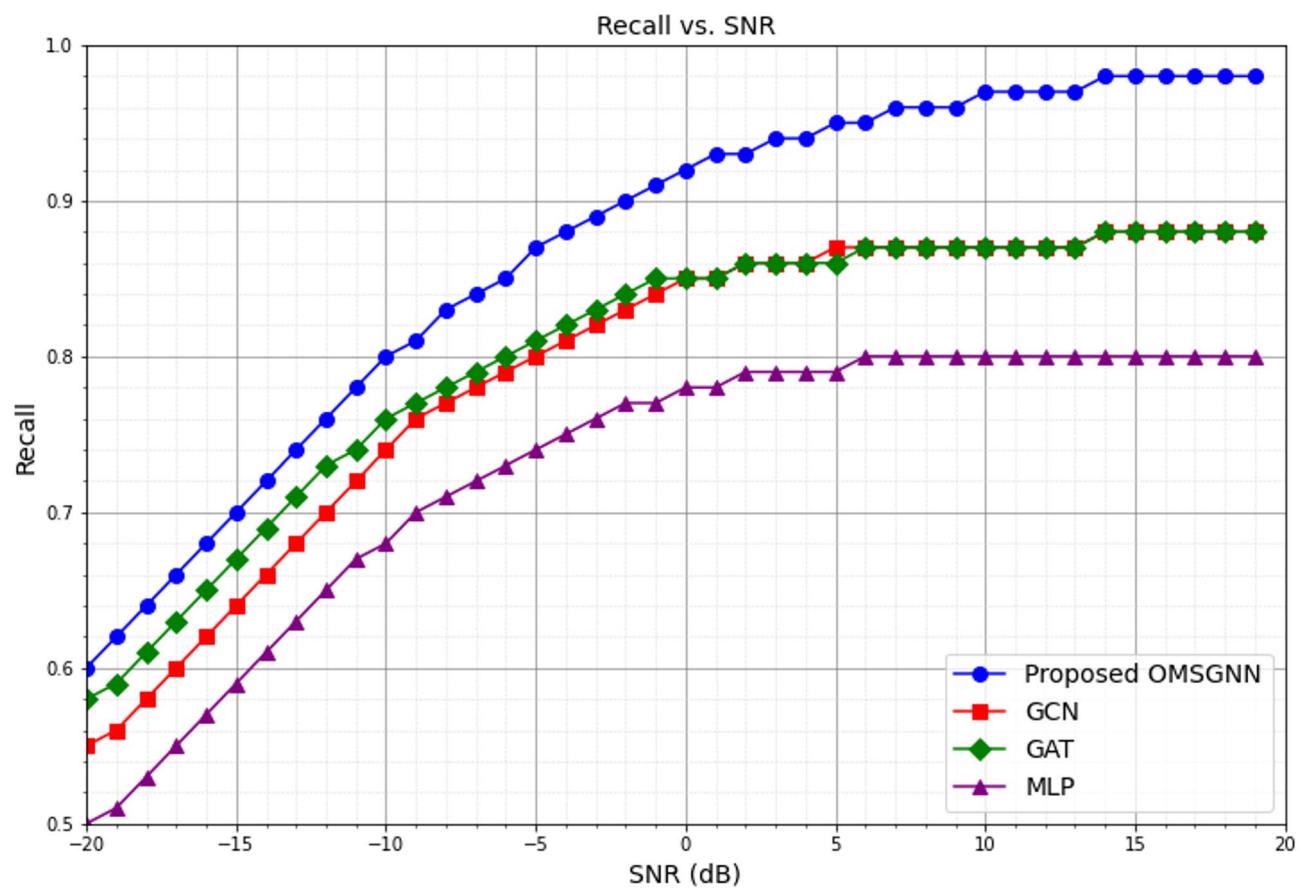
This study introduces an Optimized Multi-Scale Graph Neural Network with Attention (OMSGNNA) for cooperative spectrum sensing in cognitive radio networks operating under low signal-to-noise ratios (SNRs) and complex modulation schemes. By encoding observations as graphs, the model captures both spatial and temporal dependencies, while dual attention layers highlight the most informative features and multi-scale pooling yields resilient feature aggregation. Model parameters are further refined with Adaptive Butterfly Optimization enhanced by Lévy flights (ABO-LF), boosting sensing accuracy. Experiments on the RadioML 2016.10b dataset show that OMSGNNA reaches 98% detection accuracy at high SNRs and cuts missed detections in low-SNR scenarios by up to 30% compared with GCN, GAT, and MLP baselines. The approach overcomes the chief shortcomings of prior methods—namely their limited robustness to noise and difficulty modeling intricate spatio-temporal patterns—though its multi-scale pooling and attention layers do raise computational cost. Future work can explore lightweight variants and more energy-efficient optimization strategies.



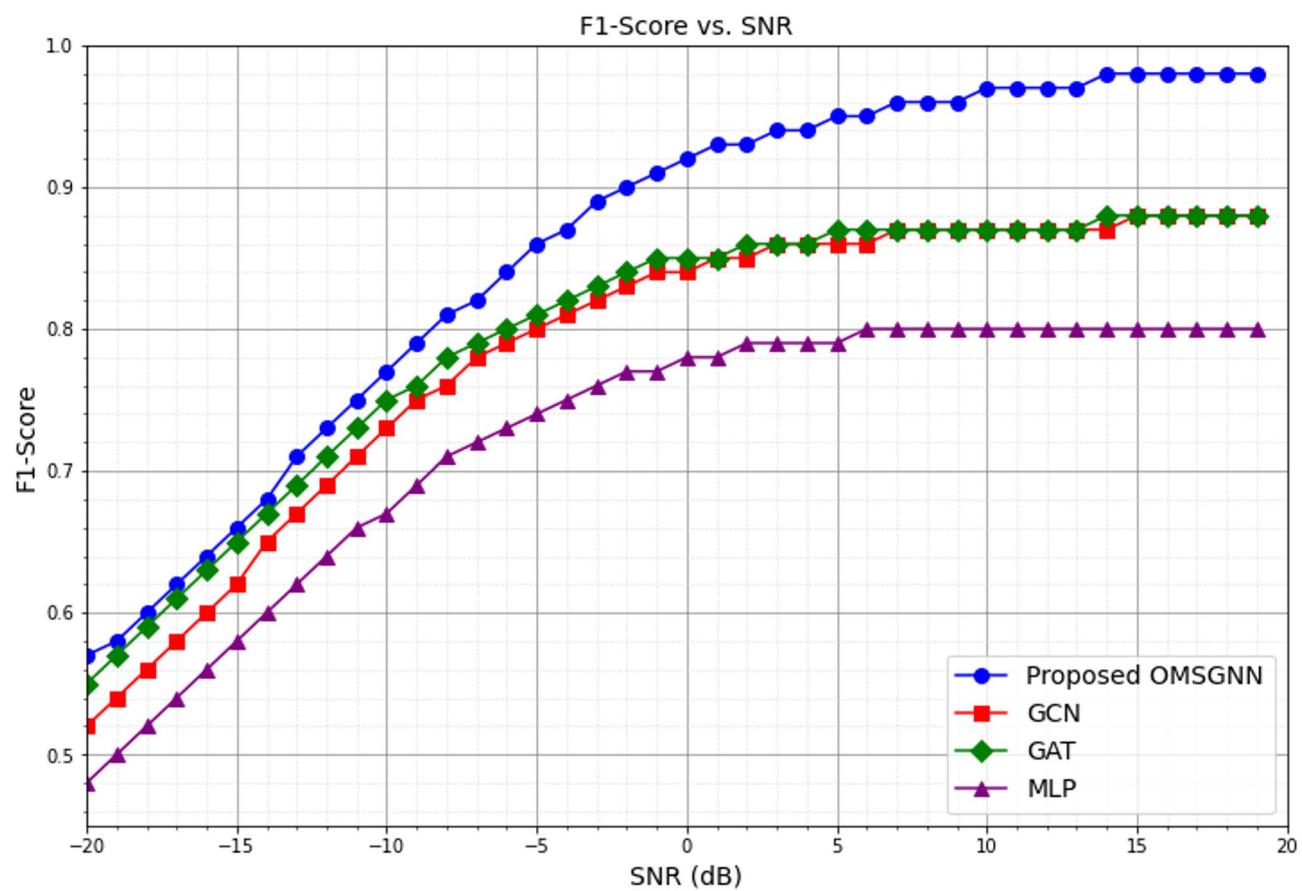
**Fig. 6.** Missed detection rate for different modulation schemes.



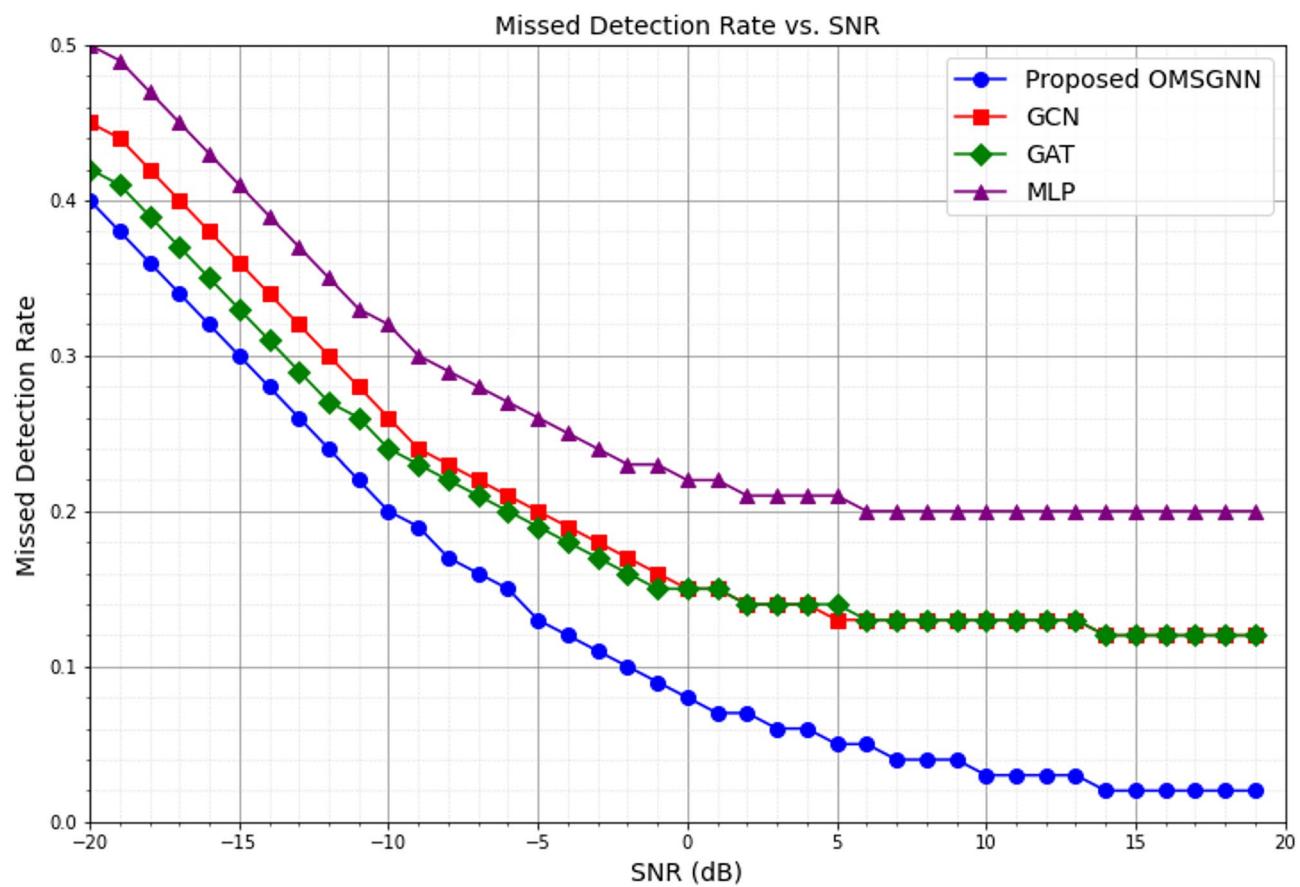
**Fig. 7.** Precision analysis.



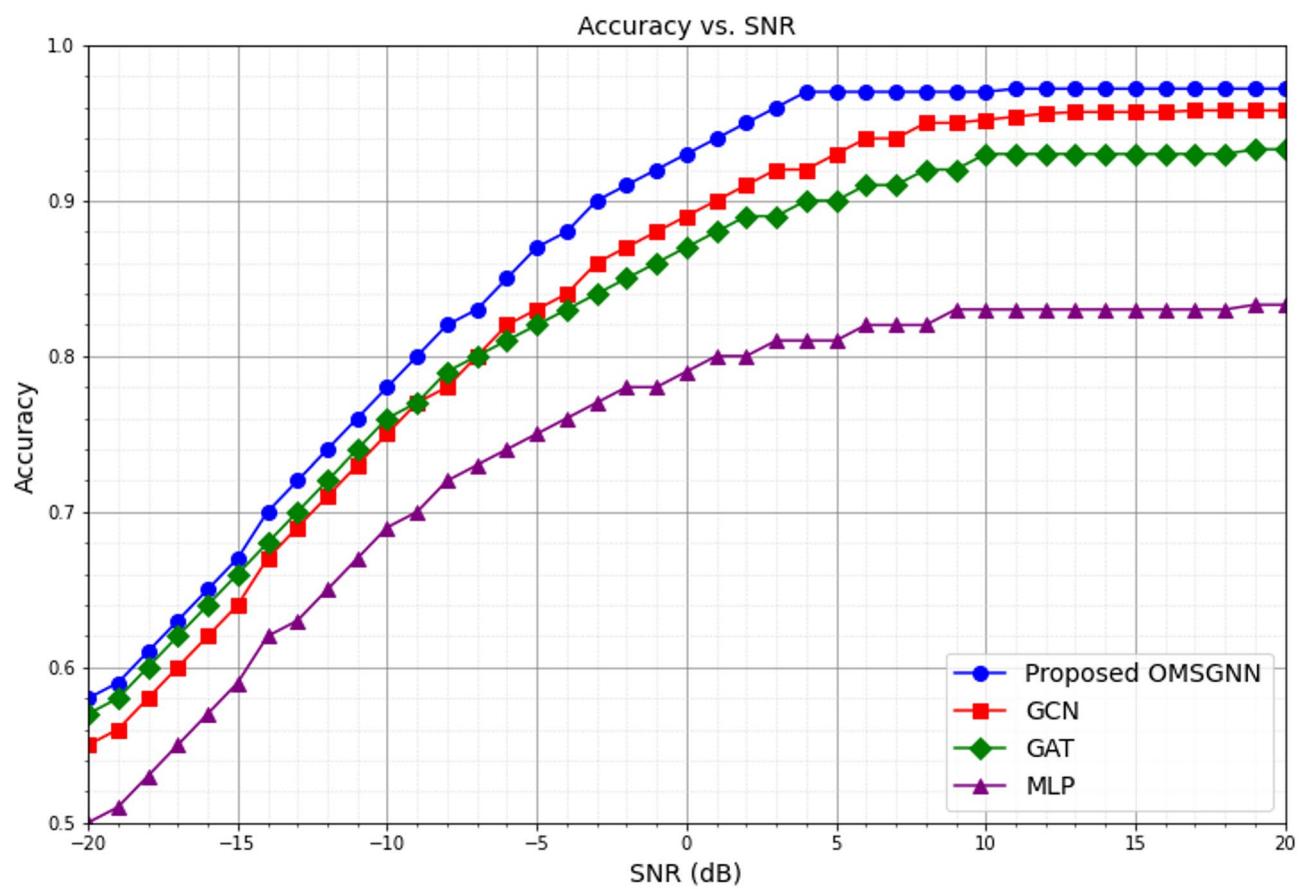
**Fig. 8.** Recall analysis.



**Fig. 9.** F1-score analysis.



**Fig. 10.** Missed detection rate analysis.



**Fig. 11.** Comparison of detection accuracy metrics.

## Data availability

The datasets generated and/or analysed during the current study are available in the Kaggle repository, <https://www.kaggle.com/datasets/marwanabudeeb/rml201610b>.

Received: 15 July 2025; Accepted: 16 October 2025

Published online: 20 November 2025

## References

1. Vidya, E.V. & Shylaja, B. S. CSS: Deep learning-based spectrum allocation with cooperative spectrum sensing for cognitive WSNs. *Int. J. Adv. Res. Ideas Innov. Technol.* **8** (2), 165–173 (2022).
2. Roopa, V. & Pradhan, H. S. Deep learning based intelligent spectrum sensing in cognitive radio networks. *IETE J. Res.* **70** (12), 8425–8445 (2024).
3. Khamayseh, S. & Halawani, A. Cooperative spectrum sensing in cognitive radio networks: A survey on machine learning-based methods. *J. Telecommun. Inform. Technol.* **81** (3), 36–46 (2020).
4. Manoharan, J. S., Vijayasekaran, G., Gugan, I. & Priyadarshini, P. N. Adaptive forest fire optimization algorithm for enhanced energy efficiency and scalability in wireless sensor networks. *Ain Shams Eng. J.* **16**, 7. <https://doi.org/10.1016/j.asej.2025.103406> (2025).
5. Samuel Manoharan, J. A novel load balancing aware graph theory-based node deployment in wireless sensor networks. *Wireless Pers. Commun.* **128**, 1171–1192 (2023).
6. Singh, S. & Kumar, S. Machine learning for cooperative spectrum sensing and sharing: A survey. *Trans. Emerg. Telecommunication Technol.* **33** (1), 1–28 (2022).
7. Solanki, S., Dehalwar, V. & Choudhary, J. Deep learning for spectrum sensing in cognitive radio. *Symmetry* **13** (1), 1–15 (2021).
8. Zheng, S. et al. Spectrum sensing based on deep learning classification for cognitive radios. *China Commun.* **17** (2), 138–148 (2020).
9. Samuel Manoharan, J. Double attribute-based node deployment in wireless sensor networks using novel weight-based clustering approach. *Sādhanā* **47** (3), 1–11 (2022).
10. Janu, D. et al. Deep learning-based spectrum sensing for cognitive radio applications. *Sensors* **24**, 24 (2024).
11. Noureddine, E., Madini, Z. & Zouine, Y. A review of deep learning techniques for enhancing spectrum sensing and prediction in cognitive radio systems: approaches, datasets, and challenges. *Int. J. Comput. Appl.* **46** (2), 1104–1128 (2024).
12. Nair, R.G. & Narayanan, K. Cooperative spectrum sensing in cognitive radio networks using machine learning techniques. *Appl. Nanosci.* **13**, 2353–2363 (2023).
13. Rabiu, E. O., Akande, D. O., Adeyemo, Z. K., Akanbi, I. A. & Obanisola, O. O. Three-state hidden Markov model for spectrum prediction in cognitive radio networks. *ABUAD J. Eng. Res. Dev. (AJERD)* **7** (2), 425–435 (2024).
14. Rabiu, E. O. et al. Three-state non-stationary hidden Markov model for an improved spectrum inference in cognitive radio networks. *J. Electr. Electron. Eng.* **13**(1), 46–58 (2025).
15. Venkatapathi, P. et al. Cooperative spectrum sensing performance assessment using machine learning in cognitive radio sensor networks. *Eng. Technol. Appl. Sci. Res.* **14** (1), 12875–12879 (2024).
16. Hossain, M. S. & Miah, M. S. Machine learning-based malicious user detection for reliable cooperative radio spectrum sensing in cognitive radio-internet of things. *Mach. Learn. Appl.* **5**, 1–9 (2021).
17. Ahmed, R., Chen, Y., Hassan, B., Du, L. & CR-IoTNet. Machine learning based joint spectrum sensing and allocation for cognitive radio enabled IoT cellular networks. *Ad Hoc Netw.* **112**, 1–26 (2021).
18. Giri, M. K. & Majumder, S. Cooperative spectrum sensing using extreme learning machines for cognitive radio networks. *IETE Tech. Rev.* **39** (3), 698–712 (2022).
19. Srivastava, V. et al. Performance enhancement in clustering cooperative spectrum sensing for cognitive radio network using metaheuristic algorithm. *Sci. Rep.* **13**, 1–14 (2023).
20. Shrotri, S.B. & Poshattiwar, S. D. An efficient cooperative spectrum sensing method using Renyi entropy weighted optimal likelihood ratio for CRN. *J. Telecommun. Inform. Technol.* **93** (3), 41–48 (2023).
21. Gao, A. et al. A cooperative spectrum sensing with multi-agent reinforcement learning approach in cognitive radio networks. *IEEE Commun. Lett.* **25** (8), 2604–2608 (2021).
22. Nair, R.G. & Narayanan, K. Cooperative spectrum sensing in cognitive radio networks using machine learning techniques. *Appl. Nanosci.* **13**, 2353–2363 (2023).
23. Chauhan, P. et al. Cooperative spectrum Prediction-Driven sensing for energy constrained cognitive radio networks. *IEEE Access* **9**, 26107–26118 (2021).
24. Uppala, A.R. Improved convolutional neural network based cooperative spectrum sensing for cognitive radio. *KSII Trans. Internet Inform. Syst. (TIIS)* **15** (6), 2128–2147 (2021).
25. Mishra, Y. & Chaudhary, V. S. Deep learning approach for co-operative spectrum sensing under congested cognitive IoT networks. *J. Integr. Sci. Technol.* **12**, 4 (2024).
26. Mondal, S., Dutta, M. P. & Chakraborty, S. K. A hybrid deep learning model for efficient spectrum sensing in cognitive radio. *J. Electr. Syst.* **20** (3), 1–14 (2024).
27. Xing, H. et al. Spectrum sensing in cognitive radio: A deep learning-based model. *Trans. Emerg. Telecommunication Technol.* **33** (1), 1–17 (2021).
28. Goyal, S. B., Bedi, P. & Vijaykumar Varadarajan. Deep learning application for sensing available spectrum for cognitive radio: an ECRNN approach. *Peer-to-Peer Netw. Appl.* **14**, 3235–3249 (2021).
29. Obite, F., Usman, A. D. & Okafor, E. An overview of deep reinforcement learning for spectrum sensing in cognitive radio networks. *Digit. Signal Proc.* **113**, 1–18 (2021).
30. Xu, M., Yin, Z., Zhao, Y. & Wu, Z. Cooperative spectrum sensing based on multi-features combination network in cognitive radio network. *Entropy* **24** (1), 1–15 (2022).
31. Fang, X. et al. CNN-transformer-based cooperative spectrum sensing in cognitive radio networks. *IEEE Wirel. Commun. Lett.* **14** (5), 1576–1580 (2025).
32. Liu, S. et al. Double-RIS assisted cooperative spectrum sensing for cognitive radio networks. *IEEE Wirel. Commun. Lett.* **14** (4), 1259–1263 (2025).
33. Li, Y., Lu, G., Ye, Y., Chen, G. & Feng, J. Cooperative spectrum sensing using weighted graph sparsity. *IEEE Commun. Lett.* **29** (2), 403–407 (2025).
34. <https://www.kaggle.com/datasets/marwanabudeeb/rml201610b>.

## Author contributions

Mrs. P.Nandhini – Research proposal – construction of the work flow and model – Final Drafting – Survey of Existing works – Improvisation of the proposed model; Dr. S. Vimalnath – Initial Drafting of the paper – Collection of datasets and choice of their suitability – Formulation of pseudocode.

## Declarations

### Competing interests

The authors declare no competing interests.

### Additional information

**Correspondence** and requests for materials should be addressed to P.N.

**Reprints and permissions information** is available at [www.nature.com/reprints](http://www.nature.com/reprints).

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025