Project Report

(For Research / Freelance / Startup Project)

School of Engineering and Applied Science (SEAS)
Ahmedabad University

Project Title: Spectrum Sensing

Course: Wireless Communication (ECE 310)

Instructor: Prof. Dhaval K. Patel

Group Number: 08

Team Members:

Vaishnavi Katba (AU2340048)
Renee Vora (AU2340059)
Sneha Mirani (AU2340071)
Pushti Sonak (AU2340082)

Date: November 16, 2025

1. Abstract

Accurate weak signal detection is one of the most basic yet challenging problems in wireless communications and radar. Herein, this article presents a structured pipeline that combines Sample Covariance Matrices (SCM) with Graph Neural Networks (GNNs), which enables robust signal classification. First, raw I/Q data is divided into time windows. Then, each window is transformed into the covariance matrix form to reflect temporal correlations, mapped as nodes in a graph where highly correlated nodes are connected by edges. Following that is the development of a specialized GNN architecture, the SCM-GNN, using Chebyshev graph convolutions with SAGPooling, trained to distinguish between the window containing the signal and the window containing only noise. The pipeline is evaluated under various SNR conditions, showing remarkable performance in terms of detection probability and AUC. The proposed approach blends statistical features of the signal with graph-based relational learning at low signal-to-noise ratios, thus yielding a scalable, interpretable method for identifying signals in extremely low-SNR scenarios.

2. Introduction

Signal detection in noise conditions is one of the most challenging and important issues in cognitive radio, radar, and wireless communications. Traditional signal detection methods based on energy detection or matched filtering usually have poor performance when the SNR is very low. Motivated by this, the paper proposes a graph-based deep learning approach to learn a mapping from the received signal representation that preserves both statistical and relational characteristics.

I/Q sample acquisition is the first step in the pipeline, which is segmented into fixed-length windows; for each of these windows, one computes an SCM to extract temporal correlations between the delayed samples. Each SCM is then represented as a graph where the matrix elements are the nodes, and the edges connect the most similar nodes according to cosine similarity. This representation naturally embeds structure and dependencies in the data that might be ignored by traditional vector-based methods.

It utilizes a GNN architecture to perform classification, incorporating Chebyshev Convolutions that extract local graph features, SAGPooling to reduce the size of the graph but preserve significant nodes, and fully connected layers that provide final prediction. The SCM-GNN is trained on classifying windows as signal-present, H_1, or noise-only, H_0, and demonstrates robust performance across a large span of SNRs. This work shows that by combining covariance-based features and graph-based learning, an effective framework for signal detection in demanding conditions is achieved that provides insights into both performance and interpretability.

3. Objectives

The main objectives of this work are:

1. To develop a structured pipeline for weak signal detection that combines Sample Covariance Matrices (SCM) and Graph Neural Networks (GNNs).

2. To transform raw I/Q data into covariance matrices and represent them as graphs, capturing temporal correlations and relational structure.

3. To design a specialized SCM-GNN architecture employing Chebyshev graph convolutions and SAGPooling for effective feature extraction and graph reduction.

4. To classify signal-present (H_1) and noise-only (H_0) windows accurately, even under low signal-to-noise ratio (SNR) conditions.

5. To evaluate the pipeline across different SNR levels using performance metrics such as detection probability (P_d) and area under the ROC curve (AUC).

6. To demonstrate that the combination of covariance-based statistical features and graph-based relational learning provides a scalable, interpretable, and robust framework for signal detection in noisy environments.

4. Methodology

We have generated the .dat files synthetically in the same format as that of the actual USRP devices. We have also used a ready made FM radio signals file which we did in our Hands On sessions.

4.1. Data Generation Parameters

The generation of .dat file consists of segments of H0 (noise-only) and H1 (signal + noise), which capture practical RF conditions, where the presence of signal fluctuates. The basic procedure involves the generation of QPSK-modulated baseband signals, passing them through a multi-antenna channel along with spatial correlation and lastly, combining them with noise at different SNRs. This is exported into binary .dat files.
We have assumed the same channel, signal assumptions and simulation parameters as the base paper [1] :

Parameter	**Value**
Antennas (M)	8
Modulation	QPSK
Symbols per sensing period	100
Samples per symbol (sps)	8
Symbol rate	1 MHz
Pulse shaping	Root Raised Cosine (RRC)
RRC roll-off	0.5
Filter span	4 symbols
Spatial correlation coefficient	$\rho = 0.5$
SNR levels generated	-20 dB, -15 dB, -10 dB, -5 dB

Table 1: Simulation parameters used to reproduce the experimental setup of the SCM-GNN paper.

- Every H1 frame generates random and energy-normalised 100 QPSK symbols, which are then mapped. To match the paper's 8 samples/symbol requirement, the symbols are upsampled by 8.

- A RRC filter (roll-off = 0.5, span = 4 symbols) is implemented. A bandlimited waveform aligned with the transmitter model is produced on convolving this filter with the QPSK samples.
- $R_{ij} = \rho^{|i-j|}$, an 8-antenna spatial correlation matrix is formed and decomposed using Cholesky. This matrix is premultiplied by signals and noise due to which a realistic antenna-to-antenna correlation is introduced.
- Hence, the H1 segments contain received signal,

  $$y_m(t) = h_m \cdot s(t) \cdot \mathrm{SNR}_{\mathrm{linear}} + n(t)$$

  where, h_m is the random complex channel gain for antenna m, $s(t)$ is the pulse-shaped transmit signal and $n(t)$ is the spatially correlated noise. The H0 consists only of the correlated noise. The SNRs for which we generated datasets are -20, -15, -10, -5 dB.
- The antenna signal is saved in a sequence of I and Q values stored as int16 numbers which is the standard format.
- We have verified the generated signals by doing basic statistics, and checked for constant values to make sure that the data still remains practical and errors can be caught.

The difference between the paper's simulation and ours is the fact that the paper's simulation works for a completely known system and clear assumptions. Whereas, we had to infer information that was not clear on the basis of our interpretation, rather than theory, because of the unavailability of clear information about the dataset that the chosen base paper uses.

4.2. SCM-GNN Model based on the Base Paper

The SCM-GNN framework operates on complex I/Q samples obtained from a multi-antenna receiver. Since real RF captures are unavailable for controlled experimentation, we synthetically generate I/Q samples that closely replicate the characteristics of actual USRP devices. Each sensing window corresponds to one short RF snapshot containing either:

- H_0: noise-only samples, or
- H_1: signal + noise samples.

4.2.1 Signal Processing

The complete signal-generation procedure is described below.

1. **QPSK Modulation-**Each H_1 frame begins with the generation of QPSK-modulated symbols. QPSK maps every pair of input bits to one of four constellation points:

   $$s[k] \in \left\{ \frac{1+j}{\sqrt{2}}, \frac{-1+j}{\sqrt{2}}, \frac{-1-j}{\sqrt{2}}, \frac{1-j}{\sqrt{2}} \right\}.$$

   The normalization by $\frac{1}{\sqrt{2}}$ ensures unit average symbol energy, which allows consistent scaling when applying different SNR levels. A block of 100 such symbols forms the basic signal sequence within each sensing window.

2. **Pulse Shaping using RRC-:** Directly transmitting QPSK impulses produces a wideband signal. To imitate a practical transmitter, we apply Root Raised Cosine (RRC) pulse shaping with roll-off factor $\alpha = 0.5$ and a span of 4 symbols. In our implementation, this filter is sampled at 8 samples per symbol (sps), matching the exact digital modulation settings used in the SCM-GNN base paper.

3. **Spatially Correlated Multi-Antenna Channel-** To simulate a realistic multi-antenna receiver, we incorporate spatial correlation across the antenna elements. The SCM-GNN paper models the correlation matrix as $R_{ij} = \rho^{|i-j|}$ where $\rho = 0.5$.

   We apply Cholesky factorization: $L = \text{chol}(R)$, which allows uncorrelated Gaussian noise to be transformed into spatially correlated noise:

   $$\mathbf{n}(t) = L \cdot \mathbf{w}(t),$$

   where $\mathbf{w}(t)$ is i.i.d. complex Gaussian noise.

4. H_1**: Signal + Noise Model** For each antenna m, the received complex baseband signal is modeled as:

   $$y_m(t) = h_m \, s(t) \, \sqrt{\text{SNR}_{\text{linear}}} + n_m(t),$$

   where: $s(t)$: pulse-shaped QPSK waveform ; h_m: complex Rayleigh fading coefficient for antenna m ; $n_m(t)$: spatially correlated noise ; $\text{SNR}_{\text{linear}} = 10^{\text{SNR}_{\text{dB}}/10}$

   This represents a typical flat-fading plus AWGN environment. Lower SNR values result in highly noise-dominated waveforms, matching the SCM-GNN paper's test range of -20 dB to 5 dB.

5. H_0**: Noise-Only Model** Under the null hypothesis, the received signal consists purely of spatially correlated noise- $y_m(t) = n_m(t)$. These windows represent idle spectrum conditions and are essential for estimating the *probability of false alarm* and for evaluating the model's ability to distinguish noise from communication signals.

4.2.2 Graph Features

Next part is to Convert RF data to Graphs. Each sensing window of the signal, has a fixed number of samples and is transformed into a Sample Covariance Matrix (SCM) to serve as a key feature for the detection of signals. The SCM captures the relationships and correlations between delayed versions of the signal within the window. In gist, by selecting a time-delay parameter, the SCM effectively measures how each sample in the window is related to other samples at different time lags.

As a result, SCM is a square matrix whose diagonal contains the variance ,which is, the power of the delayed samples. The off-diagonal elements capture the cross-correlation between different delayed samples.SCM captures both the self-similarity and the inter-dependencies of the samples of the input signal across the time axis.
By making each window a covariance matrix of this kind, we maintain important patterns in the data and decrease the dimensionality of the raw signal in a way that is effective for the GNN to process.

1. **Graph Node Features** Each entry R_{ij} of the SCM is treated as a graph node. The basic node feature is the pair of real and imaginary parts: $x_{\text{node}} = \big[\Re(R_{ij}),\ \Im(R_{ij})\big]$.

   Optionally, to provide global spectral/subspace information to the GNN, we compute the eigenvalues of R, sort them in descending orderand append the top-$k$eigenvalues to every node feature

2. **Graph Edges** We form edges based on cosine similarity between node feature vectors:
$$\text{sim}(u, v) = \frac{x_u \cdot x_v}{\|x_u\|\ \|x_v\|}.$$
   For each node we select its top-K neighbors according to $\text{sim}(\cdot, \cdot)$ and add undirected edges (implemented as bidirectional directed pairs). The resulting sparse adjacency emphasizes the strongest local statistical relations within the SCM.

3. **Neural Network Architecture** We implement the SCM-GNN architecture comprising an input linear encoder, Chebyshev spectral graph convolution,self-attention pooling (SAGPool), global readout, and a small MLP classifier. **Input projection.** The input node feature matrix $X \in \mathbb{R}^{Q\times F}$ (where $Q = \zeta^2$) is linearly projected to a hidden dimension d, followed by batch normalization and ReLU:
$$Z^{(0)} = \text{ReLU}\big(\text{BN}(XW_0 + b_0)\big).$$
   Chebyshev graph convolution (ChebConv). We adopt a Chebyshev polynomial approximation of spectral convolution. Conceptually a ChebConv layer with order K applies
$$Z^{(l+1)} = \sigma\left(\sum_{k=0}^{K} T_k(\tilde{L})\, Z^{(l)}\, \Theta_k\right),$$
   where $T_k(\tilde{L})$ is the k-th Chebyshev polynomial of the scaled Laplacian $\tilde{L}$, Θ_k are learnable parameters, and σ is a nonlinearity. Chebyshev filters provide an efficient spectral filtering mechanism without computing eigen-decompositions.

   Graph pooling (SAGPool). We apply self-attention graph pooling (SAGPool) to reduce graph size andfocus representations on the most informative nodes. SAGPool computes attention scores per node and retains the top fraction r of nodes, masking out others. Pooling improves efficiency and builds hierarchical features.

   Global readout and classification. After the convolution+pooling stack we summarize node embeddings by concatenating global mean and max pooling: $h = \text{concat}\big(\text{mean}_{\text{nodes}}(Z),\ \max_{\text{nodes}}(Z)\big)$. The vector h is passed to a two-layer MLP which outputs logits for the two classes. Applying softmax yields class probabilities: $\hat{y} = \text{softmax}(f(G))$.
   Training uses cross-entropy loss over mini-batches:
$$\mathcal{L} = -\frac{1}{B}\sum_{r=1}^{B}\big[b^{(r)}\log\hat{y}_{H_1}^{(r)} + (1 - b^{(r)})\log\hat{y}_{H_0}^{(r)}\big],$$
   where $b^{(r)} \in \{0, 1\}$ is the ground-truth label.

4.3. Patched GNN - Proposed model

The model proposed in the base paper, i.e. SCM-GNN, processes each element of the sample covariance matrix (SCM) as a node in the graph. However, it only analysis local relationships between individual matrix entries and does not capture higher level patterns in covariance matrices. Thus the paper highlights the method of patching.
A patched GNN takes the original covariance matrix and divides it into patches (small blocks). It takes each block as a super-node. Instead of just taking a single entry, it takes the block as clustered correlations, allowing it to take context as an aspect in analysis.

We have proposed a model which uses Patched GNN along with patch level attention fusion.

Key Improvements-

- Patch attention mechanism - Introduces attention weights for nodes/patches, which highlighting important graph regions and improves interpretability.
- Enhanced feature fusion- It concatenates attention summaries with pooled features rather than just using mean+max pooling.
- Multi-scale representation: Captures both local important regions (through attention) and global structural patterns (through pooling).
- Increased model capacity: Uses larger hidden dimensions (64 vs 32) for greater representational power

4.4. Evaluation Metrics

We report standard hypothesis-testing and classification metrics
Probability of Detection

$$P_d = \Pr\left(\hat{y} = 1 \mid H_1\right).$$

Empirically P_d is estimated over test H_1 windows by counting detections at a threshold.
Probability of False Alarm (Pf).

$$P_f = \Pr\left(\hat{y} = 1 \mid H_0\right).$$

- **ROC/AUC**
  ROC or Receiver Operating Characteristic curve (Also known as P_d vs P_f curve) is a visualization tool to determine the performance of a binary models, in our case, hypothesis.
  H_0 : Signal absent (noise only)
  H_1 : Signal present (signal + noise)
  ROC curse shows the trade-off between True Positive Rate (TPR) (also known as Probability of detection (P_d))and False Positive Rate (FPR) (also known as Probability of failure (P_{fa})).
  We then plot the curve by taking TPR at y-axis and FPR at x-axis. The Area under the curve summarizes the ROC performance. The higher the area (highest=1), the better.Therefore, closer the curve is to the top-left corner, the better the model's detection ability.

- Pd vs SNR.
  For each SNR we compute P_d(SNR) using a test set of H_1 windows at that SNR. The curve should always be increasing in nature.

- Accuracy In our models, accuracy measures the proportion of correctly classified covariance matrices (both H_0 and H_1 cases).

To compare models at a fixed false alarm level P_f^*, find the threshold τ such that $P_f(\tau) = P_f^*$ (from the ROC) and report the corresponding $P_d(\tau)$.

These metrics (ROC, AUC, Pd–SNR curves, and Pd at fixed Pf) are standard in spectrum sensing, which we have also used for comparison between the models.

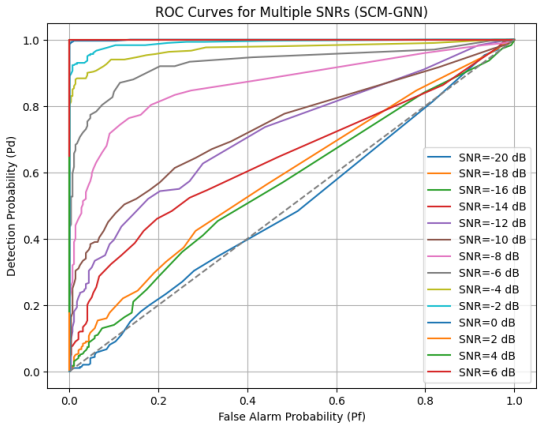
Table 2: SCM-GNN Pipeline Parameters

Parameter	**Description**	**Value**
Dataset file	Raw I/Q data	Final_5dB.dat, Final_0dB.dat, Final_5dB.dat
Sample size (window length)	Number of samples per window	100
Number of windows	Signal / Noise	800 / 800
Time-delay size (zeta)	Covariance matrix dimension	8×8
Node features	Real + Imag parts of SCM entries	2 features per node
Top-k edges	Nearest neighbors for graph	8
Graph nodes	Number of nodes per graph	64
Batch size	Training batch	200
Epochs	Number of training epochs	50
SNR range	Signal-to-noise ratio (dB)	$20 \rightarrow 6$ (step 2)
GNN hidden features	Hidden layer size	32
ChebConv K	Chebyshev polynomial order	3
Pooling	Graph pooling	SAGPooling, ratio 0.5
Output	Binary classification	Signal present (1) / Noise (0)

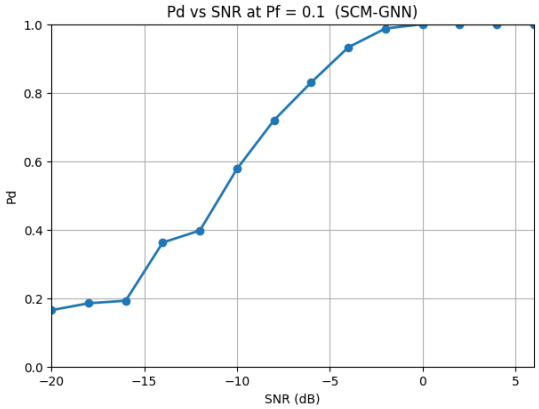
Table 3: Dataset, Graph Sizes, and SNR Settings (SCM-GNN)

Dataset	**Window Size**	**No. of Windows**	**Covariance**	**Graph Nodes**	**Top-k Edges**	**SNR**
Final_-5dB.dat	100	800	8×8	64	8	-5
Final_0dB.dat	100	800	8×8	64	8	0
Final_5dB.dat	100	800	8×8	64	8	5

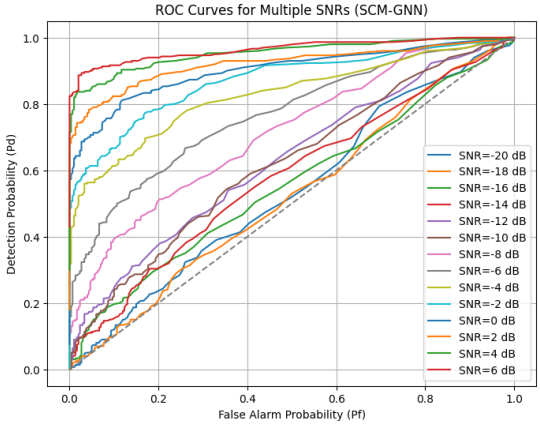
5. Results



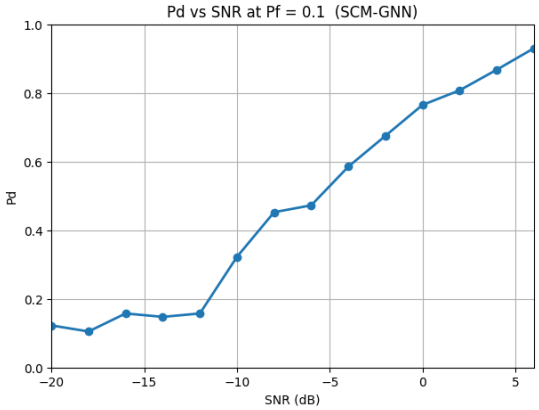
(a) FM:ROC curves



(b) FM:PD vs SNR curve

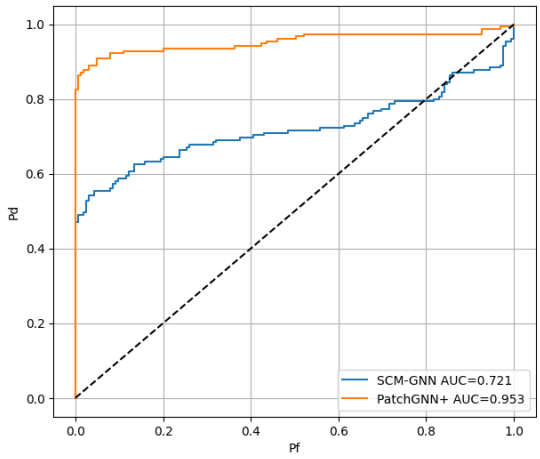


(c) Generated dataset:ROC curves

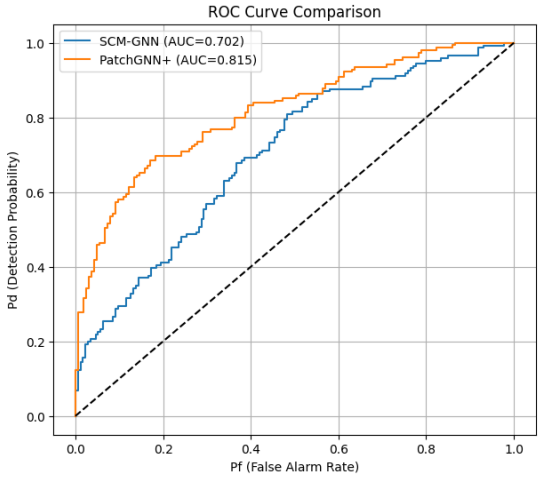


(d) Generated dataset : PD vs SNR

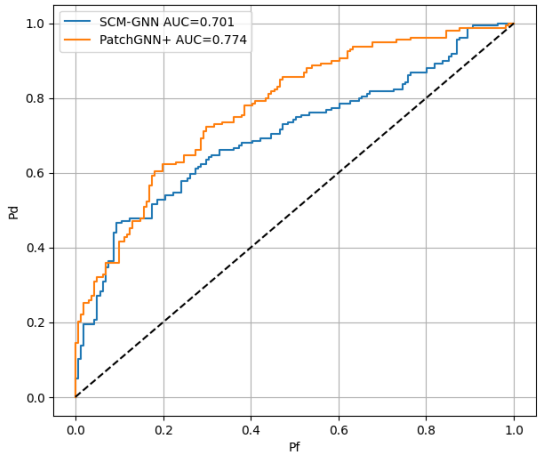
Figure 1: Reproduced SCM-GNN graphs with generated and real world dataset



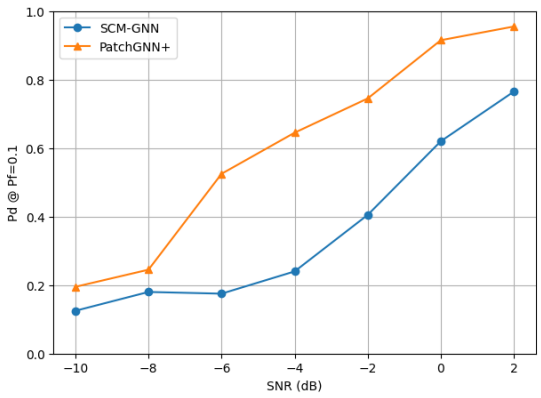
(a) SNR = -10 dB



(b) SNR = -15 dB



(c) SNR = -20 dB



(d) PD vs SNR

Figure 2: Combined ROC and PD–SNR plots in a 2×2 layout

6. Discussion

6.1. Interpretation

The simulation results shows the difference between the baseline SCM-GNN model in our chosen research paper and the proposed PatchGNN framework using multiple SNR values and ROC analyses. The PatchGNN has higher Pd values as shown in the Pd vs SNR curve at all SNR levels. This describes that PatchGNN handles noisy situations much better than the SCM-GNN model given in the research paper. It can be seen that when the SNR value is high, the PatchGNN model performs almost perfectly. This gives a gist about how PatchGNN model is able to use the spatial and spectral information in the data more effectively.

The ROC curves also support these observations. As we know the AUC value more closer 1, denotes a better model. At -20dB, -15dB, and -10dB, PatchGNN gives higher AUC values compared to SCM-GNN. This highlightes the effectiveness of the patch-based graph neural networks and enhances the strategy in noisy environments. Overall, the results show that PatchGNN performs effectively in low-SNR values. However some important observations and drawbacks -

- As you lower the SNR , the Patched GNN is still working better , but still drobs exponentially
- Patched GNN , as a model was very easily overtrained. At one point , the model started giving higher AUC at lower SNR and Medium level AUC at at high SNR
- The dataset used mattered a lot as FM dataset gave better outputs than the generated dataset.

6.2. Challenges faced:

- In reproducing the base paper, the foremost challenge was to generate the dataset it used. It was not publicly available, and used custom dataset with no clear details about the steps involved in the generation of covariance matrices and pre-processing.
- We attempted multiple times to generate a dataset similar to that of the base paper, but the .dat files did not align to the base paper's format.
- The synthetic dataset generated mostly contained invalid float32 sequences or only zeros, which were unusable.
- We later tried the implementation using the real dataset generated using the USRP device. It had a low number of usable windows from the large sample, which was not enough for GNN. Also., inferring to a suitable window ssize without any reference was difficult.
- Overall, combining the evaluation of ROC/AUC curves along with GNN, above generating the dataset repeatedly was challenging.

7. Conclusion

The contribution of this work was to provide a structured pipeline that combined SCM with GNNs to help in the robust weak signal detection in noisy environments. This approach effectively transforms raw I/Q data into covariance matrices, represents them as graphs, and extracts meaningful graph features by leveraging a customized SCM-GNN architecture with Chebyshev convolutions and SAGPooling. Experimental results over various SNR levels validate that the proposed pipeline achieves a high detection probability and robust ROC performance, even under extremely low SNR. The approach combines statistical signal features with graph-based relational learning; thus, it offers a scalable, interpretable, and efficient signal classification framework.

In all, this work has demonstrated the effectiveness of covariance-based signal representation combined with graph neural networks for enhancing performance in difficult detection tasks. It therefore points to a promising direction for future studies in cognitive radio, radar, and other wireless communication applications.

References

[1] Y. Dong, M. Zhang, X. Cheng, and H. Wang, "SCM-GNN: A Graph Neural Network-Based Multi-Antenna Spectrum Sensing in Cognitive Radio," *IEEE Transactions on Cognitive Communications and Networking*, vol. 11, no. 1, pp. 127–144, Jul. 2024, doi: 10.1109/tccn.2024.3431923.

[2] Y. Zhang et al., "SpecKriging: GNN-Based Secure Cooperative Spectrum Sensing," *IEEE Transactions on Wireless Communications*, vol. 21, no. 11, pp. 9936–9946, Nov. 2022, doi: 10.1109/twc.2022.3181064.

[3] Y. Li, G. Lu, and Y. Ye, "Spectrum Sensing Based on Graph Weighted Aggregation Operator," *IEEE Communications Letters*, vol. 27, no. 11, pp. 3132–3136, Sep. 2023, doi: 10.1109/lcomm.2023.3314805.

[4] J. Lin et al., "An Overview of Challenges and Requirements for Real-Time Spectrum Sensing in Modern RF Autonomy Systems," *IEEE Design & Test*, pp. 1–1, 2025, doi: 10.1109/mdat.2025.3594311.

[5] S. Wu, G. Hu, and B. Gu, "Spectrum-sensing algorithm based on graph feature fusion," *IET Radar, Sonar & Navigation*, Dec. 2024, doi: 10.1049/rsn2.12674.

[6] Y. Dong et al., "Multi-View Graph Neural Networks for Spectrum Sensing in Cognitive Radio," *IEEE Internet of Things Journal*, pp. 1–1, Jan. 2025, doi: 10.1109/jiot.2025.3569323.

[7] X. Zhang et al., "SAMS-GNN: Self-Adaptive Multi-Scale Graph Neural Network for Multi-Band Spectrum Prediction," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, Jan. 2024, doi: 10.1109/tccn.2024.3483202.

[8] S. Wang, X. Wang, K. Sun, S. Jajodia, H. Wang, and Q. Li, "GraphSPD: Graph-Based Security Patch Detection with Enriched Code Semantics," in *Proc. IEEE Symposium on Security and Privacy*, 2023, doi: 10.1109/SP46215.2023.10179479.