School of Engineering and Applied Sciences (SEAS)
Ahmedabad University, Ahmedabad

Project Assessment - 1

Spectrum Sensing

Domain: Research

Course:	ECE310 – Wireless Communications	
Course Instructor:	Prof. Dhaval Patel	
Project Mentor:	Prapti Patel	
Team Members:	**Group 8 (G8)**	
	Vaishnavi Katba	AU2340048
	Renee Vora	AU2340059
	Sneha Mirani	AU2340071
	Pushti Sonak	AU2340082

October 17, 2025

Contents

1. List 5–10 relevant and high-quality research papers related to the domain of spectrum sensing. Provide a 1-paragraph summary for at least three of the most relevant ones, including their key contributions and limitations.

1.1 SCM-GNN: A Graph Neural Network-Based Multi-Antenna Spectrum Sensing in Cognitive Radio

Youqiang Dong, Min Zhang, Xicheng, and Hai Wang
IEEE Transactions on Cognitive Communications and Networking, Vol. 11, No. 1, Feb. 2025

Summary: This paper [1] has proposed a spectrum sensing method that detects the presence of a primary user(PU) signal in a wireless channel and tests it under various realistic scenarios. It is built upon forming a covariance matrix from received signals and has considered a smoothing factor in it. To learn PU detection patterns in these signals, a graph neural network(GNN) approach is applied. This method is tested under noise uncertainty, large-scale fading, a varied number of antennas and under imperfect reporting channels, and its performance is compared with popular existing methods. It gives fairly accurate spectrum sensing results in real-world wireless conditions.

Key contributions and limitations:

- This method doesn't require knowing the signal structure of the PU, but only some historical sensing data. It performs fairly well under practical conditions, which have been validated using the actual USRP hardware. Furthermore, it is scalable and doesn't require retraining for different channels.
- The model requires a training phase or even re-training under drastic channel conditions as its performance degrades. It is computationally heavy, involving GNN with matrix operations. Moreover, data collection is difficult because of the device limitations for multi-antenna spectrum sensing.

1.2 SpecKriging: GNN-Based Secure Cooperative Spectrum Sensing

Yan Zhang, Dianqi Han, Ang Li, Jiawei Li, Tao Li, Rui Zhang, and Yanchao Zhang
IEEE Transactions on Wireless Communications, Vol. 21, No. 11, Nov. 2022

Summary: The method suggested by this paper [2] applies GNN to learn patterns from the data acquired from sensors placed at different regions. The sensors together capture radio signals, which could be wrong and may give unreliable results. GNN learns the behaviour of signals across locations, and uses the safe data (from anchor sensors) that is reliable as a reference. In case of detection of suspicious data, the model ignores it, and the GNN is retrained using clean data to make reliable spectrum sensing decisions.

Key contributions and limitations:

- SpecKringing method is fast, efficient, scalable and gives accurate results for both homogeneous and heterogeneous sensor systems as compared to the Ordinary kringing method.
- This method initially needs a set of secure sensors and retraining to start with, and assumes that PUs are stationary and that the sensors' locations are known. Thus, it underperforms when these assumptions are not met. It works only for data of signal strength type and requires manual threshold setup for unreliable data.

1.3 Spectrum Sensing Based on Graph Weighted Aggregation Operator

Yuxin Li, Guangyue Lu, and Yinghui Ye
IEEE Communications Letters, Vol. 27, No. 11, Nov. 2023

Summary: The approach proposed by this paper [3] combines the graph structure and the graph signal to form a test statistic that evaluates whether PU exists or not. The model converts signals into a power spectrum, which is then modelled to a weighted graph that has quantised frequency points as nodes, edges connect the frequency nodes that change together and the edge weights represent how strongly the nodes are related. A decision threshold (from Monte Carlo trials) is applied to the Graph Weighted Average Detector(GWAD); if the calculated test statistic score is lower than the threshold, PU is present; else PU is absent.

Key contributions and limitations:

- This method gives higher accuracy at low SNR values than traditional methods. At moderate SNR values, it gives almost perfect detection. It considers both graph structure and power information, thus giving better graph conversion.
- The model assumes no fading in the channel, which is unrealistic and considers only AWGN noise, which is an ideal type of noise. The method hasn't been extended to multiple Secondary Users. The threshold is not adaptive for real-time conditions.

1.4 An Overview of Challenges and Requirements for Real-Time Spectrum Sensing in Modern RF Autonomy Systems

Jiahao Lin, H. Umut Suluhan, Hyunwon Chung, Arindam Dutta, Anish Vipperla, Gerard Gubash, Jacob Holtom, Bernd-Peter Paris, Chaitali Chakrabarti, Daniel W. Bliss, David

Blaauw, Hun-Seok Kim, Ali Akoglu, Umit Y. Ogras
IEEE Design & Test, 2025

1.5 Spectrum-Sensing Algorithm Based on Graph Feature Fusion

Shanshan Wu, Guobing Hu, Bin Gu
IET Radar, Sonar & Navigation, 2024

1.6 Multiview Graph Neural Networks for Spectrum Sensing in Cognitive Radio

Youqiang Dong, Min Zhang, Yu Huo, Xi Cheng, Chenyu Gao, Jingwen Wang, Hai Wang
IEEE Internet of Things Journal, 2025

1.7 SAMS-GNN: Self-Adaptive Multi-Scale Graph Neural Network for Multi-Band Spectrum Prediction

Zhang, Xile and Peng, Yang and Huang, Hao and Wang, Yu and Wang, Qin and Lin, Yun and Qin, Zhijin and Gui, Guan
IEEE Transactions on Cognitive Communications and Networking, June 2025

Other relevant studies include Lin *et al.* [4], Wu *et al.* [5], Dong *et al.* [6], Zhang *et al.* [7].

2. Discuss how the insights or limitations identified in your reviewed papers directly influence your proposed research design.

- The SCM-GNN model [1] has used smoothed covariance matrix as the input through which the GNN model learns structural relationships with antennas and detects the primary user in noisy and fading channels. It requires retraining under drastic channel conditions.

  Our proposed model uses covariance matrix input with the smooth factor of $\zeta = 8$. The GNN is trained with Rayleigh fading and noise uncertainty ($\sigma\epsilon[0.6, 1.4]$) to reduce retraining needs and to generalize across various channel. We have used training stabilisers to make the training less prone to errors.

- SpecKriging method [2] works well for cooperative sensing and uses anchor sensors to detect suspicious sensors and improve data reliability, but works upon a few assumptions.

The design we have implemented uses attention-based weighting, which automatically gives less importance to unreliable antennas. In our testing, we break the above assumptions and consider that the signal sources move and thresholds are learned to adapt.

- The GWAD [3] gives a strong signal detection at low SNRs, but is unrealistic considering only AWGN fading without cooperative sensing and Monte Carlo thresholding.

  In our design, we have extended AWGN to Rayleigh fading and implemented adaptive thresholding as mentioned above. We have added similar features to GWAD, like energy, variance and correlation.

Hence, the insights and the limitations from the reviewed papers influence our research design. It very well considers practical scenarios, accuracy, performance needed and adaptability to real-time wireless conditions.

3. Clearly state the specific research gap you aim to address and explain how your problem statement evolved from prior work.

Most spectrum sensing methods like Energy Detection (ED), Maximum Eigenvalue Detection (MED), Graph Convolutional Network (GCN), Maximum Eigenvalue–Arithmetic Mean (ME-AM), depend on complex models that are not adaptable to real-world scenarios and involve high computational complexity. They are based on prior information about PUs and lose accuracy in detection in practical wireless conditions.

3.1 Research gaps:

- In earlier works, feature design is restricted to energy values, covariance data, signal strength and not any statistical or spectral feature which may improve detection at lower SNRs.
- Models have addressed latency and complexity, but fast models are demanded for real-time detection.
- Methods that require training and have ideal or static channel assumptions have lesser adaptability and are unrealistic.
- Lacked methods to control the unreliable sensing data.

3.2 Problem statement evolution:

Our research focuses on comparing a Spatial Covariance Matrix based GNN (SCM-GNN) and a Convolutional Neural Network (CNN) for realistic spectrum sensing under different

SNR values, fading conditions like AWGN and Rayleigh fading and SpecKriging preprocessing.
Overall, this approach aims to improve reliability through pre-processing before training. This will handle uncertainty and stabilise decision-making. In addition to that, our model integrates the strength of graph neural learning and convolutional processing.

4. Explain what would be the expected outcome or measurable improvement if your problem is successfully solved in the context of wireless communications.

This work is mainly aimed at applying and comparing a Graph Neural Network (SCM-GNN) and Convolutional Neural Network (CNN) to spectrum sensing in cognitive radio systems. The models are intended to determine the availability of a primary user signal with a high level of accuracy in the presence or absence of the Signal-to-Noise Ratios (SNR) with a high degree of accuracy.

In the event that the proposed system operates successfully, it is envisaged that the provision of the cognitive radio will increase its capabilities of detecting spectrum holes (free frequency bands) at low SNR levels. The SCM-GNN model is expected to be superior to the CNN and achieve success in capturing spatial and relational interdependence between the multiple antennas, instead of flat covariance matrix image as the input.

Following are the measurable improvements expected:

- **Higher Detection Accuracy:** The GNN should reliably detect the presence of a signal, even when it is weak or distorted by noise.
- **Improved ROC-AUC Performance:** The Area Under the ROC Curve (AUC) will be higher for the GNN, indicating better discrimination between signal and noise, even in low SNRs.
- **Efficient Spectrum Utilization:** More accurate detection enables secondary users to utilize free spectrum bands effectively without interfering with licensed users.

A trained GNN model will lead to more improved spectrum efficiency, and reduced interference.As a result, it contributes to smarter and more adaptive wireless communication systems.

5. Include a block diagram or flowchart illustrating your proposed method. Provide a brief explanation highlighting the novelty or uniqueness of your approach.

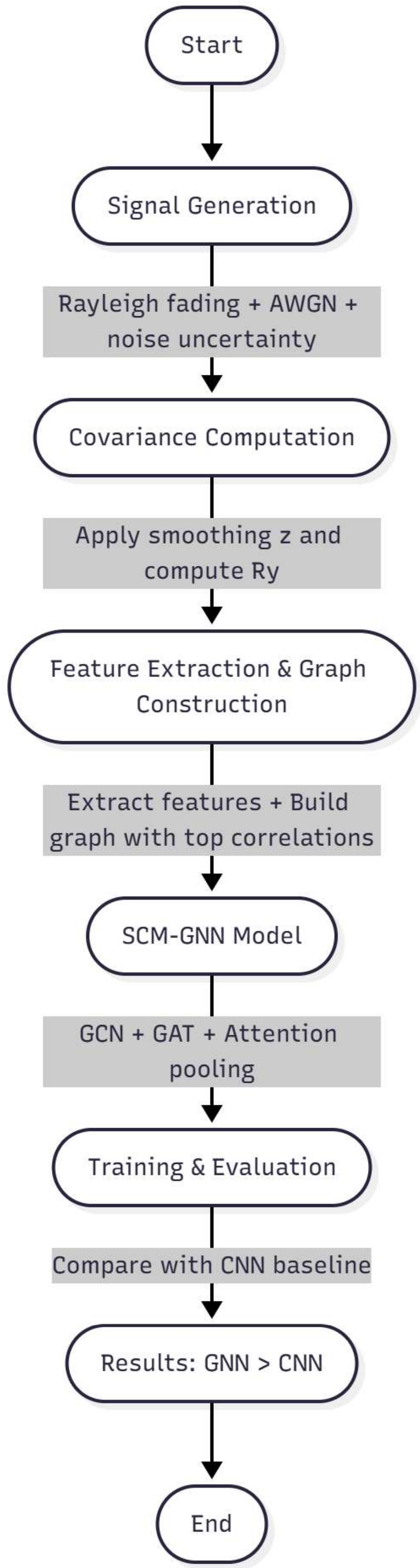


Figure 1: Proposed SCM-GNN based signal processing framework.

Explaining our approach

- The uniqueness of our approach lies in the integration of statistical signal modeling and graph-based deep learning. Signals started and subjected to Rayleigh fading, AWGN and noise uncertainty, closely running through the real-world wireless environment. Our model computes the covariance matrix, focusing on both spatial and temporal relationships within the data. Additionally, we have used the smoothing factor is applied which further filters the covariance estimate, ensuring stability and robustness against noise.

- Our approach gives a graph structure from these features, building relationships between antenna nodes based on their strongest correlations. This ensures GNN to model spatial dependencies. The SCM-GNN model integrates GCN, GAT and attention pooling, understands about the informative connections.

- Finally, the results show that our proposed SCM-GNN performs better than the traditional CNN models maintaining good performance in noisy environments, enables secondary users to utilize free bands.

6. Describe how you will evaluate the effectiveness of your proposed method. Mention metrics, benchmarks, or baseline models you plan to use.

6.1 Baseline models

AWGN-CNN baseline

We ar using a simple Convolutional Neural Network (CNN) as a baseline to compare with our proposed GNN based model. In this model, the smoothed covariance matrices are treated like small images. The CNN learns patterns of these images and decides whether a signal is present or absent.[1]

It has two convolutional layers which extracts features from the matrix (images), another layer to reduce the data size. These are fully connected layers which make the final decision. This approach is helpful for the model to learn spatial features, such as how the values in the covariance matrix vary together.

We used the CNN baseline to measure how much improvement the proposed SCM-GNN model shows over a simpler deep learning approach. Both models are trained and tested on the same dataset, under the same channel and noise conditions. Also we use the metrics explained below to determine the better one.

6.2 Metrics

Presently, We have evaluated SCM-GNN against CM-CNN using ROC/AUC , confusion matrices and Accuracy.

- **ROC/AUC**
  ROC or Receiver Operating Characteristic 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.

- **Accuracy**
  In our SCM-GNN model, accuracy measures the proportion of correctly classified covariance matrices (both H_0 and H_1 cases). It is computed from the confusion matrix after applying a fixed decision threshold (0.5). However in real world scenarios, metric such as ROC/UC and PD-SNR curves are a better metric.

We further plan on using PD vs SNR curves aswell.

7. Attach your initial simulation code (MATLAB, Python, etc.) with clear comments. Include basic graphs or plots that demonstrate your preliminary results.

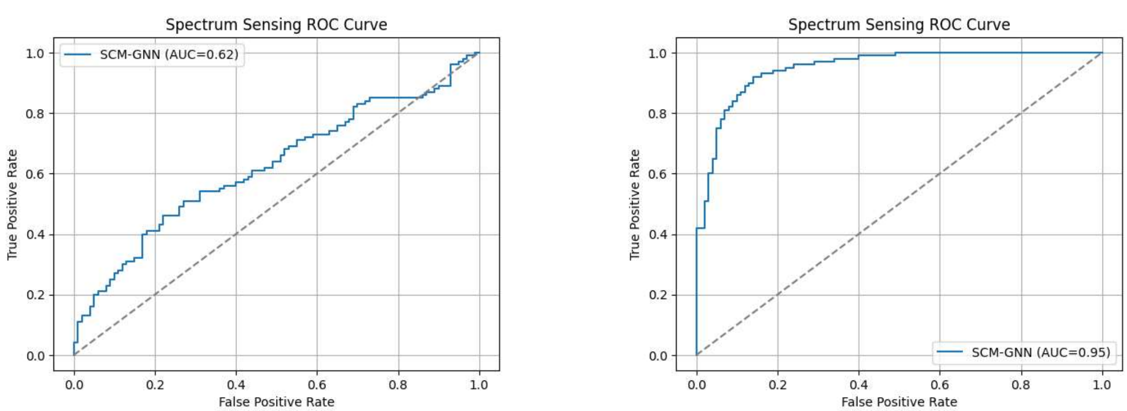
7.1 Simple GNN based Spectrum sensing code

```
# ================================
# scm-gnn simplified demo in colab
# ================================

!pip install torch torchvision torchaudio torch-geometric matplotlib
    scikit-learn -q

import torch
import torch.nn.functional as F
from torch_geometric.nn import GCNConv, global_mean_pool
import numpy as np
from sklearn.metrics import accuracy_score, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

# ------------------------------
# 1. generate synthetic data
# ------------------------------
```

```python
def generate_signals(M=8, N=1000, SNR_dB=-15, num_samples=1000):
    # make fake signals for h0 and h1
    SNR = 10 ** (SNR_dB / 10)
    X_data, y_labels = [], []

    for _ in range(num_samples):
        # random signal and noise
        s = np.exp(1j * 2 * np.pi * np.random.rand(N))
        h = (np.random.randn(M) + 1j*np.random.randn(M)) / np.sqrt(2)
        w = (np.random.randn(M, N) + 1j*np.random.randn(M, N)) / np.sqrt(2)

        # h1: signal + noise
        y1 = np.outer(h, s) + (1/np.sqrt(SNR))*w
        Ry1 = np.cov(y1)

        # h0: noise only
        y0 = (1/np.sqrt(SNR))*w
        Ry0 = np.cov(y0)

        # store data
        X_data.append(Ry1)
        y_labels.append(1)
        X_data.append(Ry0)
        y_labels.append(0)

    return np.array(X_data), np.array(y_labels)

X, y = generate_signals(num_samples=500)
print("Data shape:", X.shape)

# ------------------------------
# 2. build graph data
# ------------------------------
from torch_geometric.data import Data, DataLoader

def matrix_to_graph(R):
    # make graph from covariance matrix
    A = np.abs(R)
    np.fill_diagonal(A, 0)
    edge_index = np.array(np.nonzero(A > np.mean(A))).astype(np.int64)
    edge_index = torch.tensor(edge_index, dtype=torch.long)
    x = torch.tensor(np.real(R), dtype=torch.float)
    return Data(x=x, edge_index=edge_index)

# convert all to graphs
graph_data = [matrix_to_graph(R) for R in X]
for g, label in zip(graph_data, y):
```

```
    g.y = torch.tensor([label], dtype=torch.long)

# split train/test
split = int(0.8 * len(graph_data))
train_data, test_data = graph_data[:split], graph_data[split:]
train_loader = DataLoader(train_data, batch_size=16, shuffle=True)
test_loader = DataLoader(test_data, batch_size=16)

# ------------------------------
# 3. define gnn model
# ------------------------------
class SimpleSCMGNN(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels=32):
        super().__init__()
        self.conv1 = GCNConv(in_channels, hidden_channels)
        self.conv2 = GCNConv(hidden_channels, 16)
        self.fc = torch.nn.Linear(16, 2)

    def forward(self, x, edge_index, batch):
        # forward pass
        x = F.relu(self.conv1(x, edge_index))
        x = F.dropout(x, 0.2, training=self.training)
        x = F.relu(self.conv2(x, edge_index))
        x = global_mean_pool(x, batch)
        return self.fc(x)

# set device and optimizer
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model = SimpleSCMGNN(in_channels=X.shape[-1]).to(device)
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)

# ------------------------------
# 4. train the model
# ------------------------------
for epoch in range(40):
    model.train()
    total_loss = 0
    for batch in train_loader:
        batch = batch.to(device)
        optimizer.zero_grad()
        out = model(batch.x, batch.edge_index, batch.batch)
        loss = F.cross_entropy(out, batch.y)
        loss.backward()
        optimizer.step()
        total_loss += loss.item()
    print(f"Epoch {epoch+1}, Loss: {total_loss:.3f}")

# ------------------------------
```

```python
# 5. evaluate
# ------------------------------
model.eval()
y_true, y_pred, y_prob = [], [], []
for batch in test_loader:
    batch = batch.to(device)
    out = model(batch.x, batch.edge_index, batch.batch)
    prob = F.softmax(out, dim=1)[:, 1].cpu().detach().numpy()
    pred = out.argmax(dim=1).cpu().numpy()
    y_true.extend(batch.y.cpu().numpy())
    y_pred.extend(pred)
    y_prob.extend(prob)

# calc metrics
acc = accuracy_score(y_true, y_pred)
auc = roc_auc_score(y_true, y_prob)
print(f"\nTest Accuracy: {acc:.3f}, ROC-AUC: {auc:.3f}")

# ------------------------------
# 6. plot roc curve
# ------------------------------
fpr, tpr, _ = roc_curve(y_true, y_prob)
plt.plot(fpr, tpr, label=f'scm-gnn (auc={auc:.2f})')
plt.plot([0,1],[0,1],'--',color='gray')
plt.xlabel('false positive rate')
plt.ylabel('true positive rate')
plt.title('spectrum sensing roc curve')
plt.legend()
plt.grid(True)
plt.show()
```

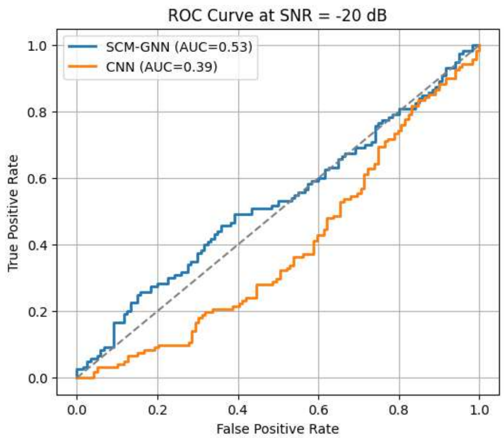


(a) Simple GNN-based output, SNR=-20

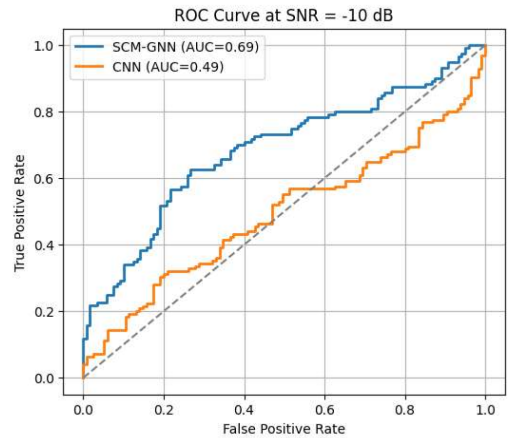
(b) Simple GNN-based output, SNR=-10

Figure 2: ROC curves of basic GNN model under different SNRs

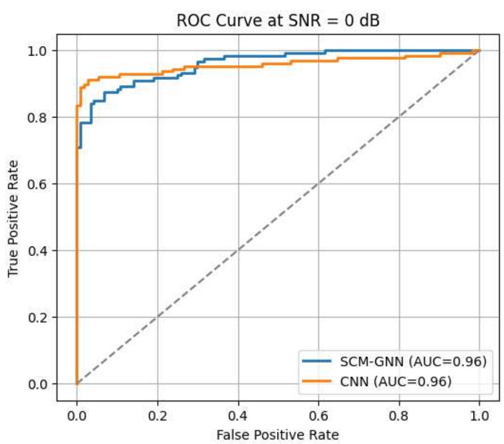
7.2 Comparing GNN based Spectrum sensing with CM-CNN based.



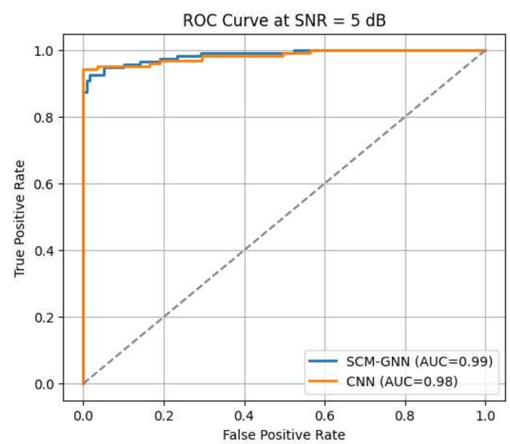
(a) ROC curves under SNR=-20dB



(b) ROC curves under SNR=-10dB



(c) ROC curves under SNR=0dB

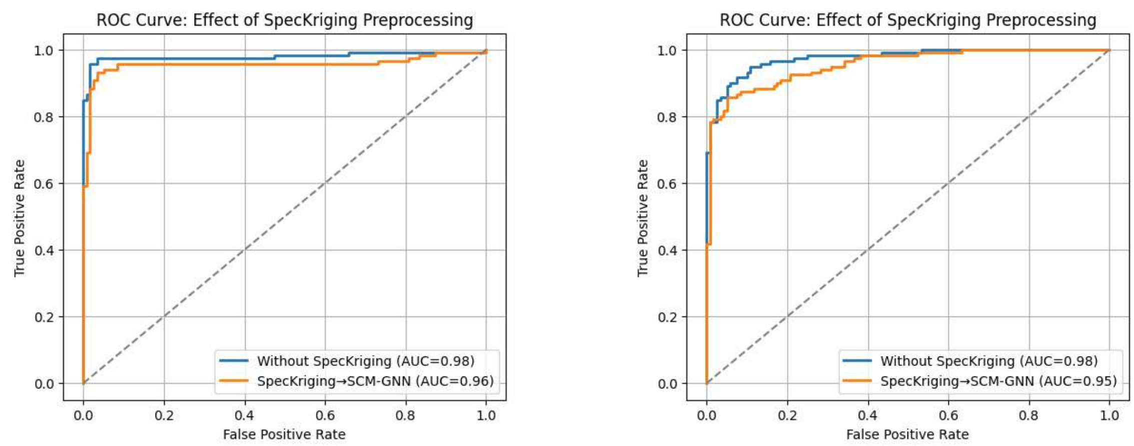


(d) ROC curves under SNR=5dB

Figure 3: ROC curves of SCM-GNN and CM-CNN models under different SNR values

7.3 Incorporating Speckriging Feature in this model

SpecKriging is a GNN based model which improves spectrum sensing through learning how nearby sensors influence each other based on their physical locations.It predicts missing or unreliable signal data and detects malicious sensors. It acts as a secure denoising layer before SCM-GNN, restoring trustworthy RSS features and improving detection accuracy.



(a) SNR=-20dB

(b) SNR=-10dB

Figure 4: ROC curves of SCM-GNN and CM-CNN models with SpecKriging preprocessing

8. Identify which parameters or factors in your simulation most strongly influence the observed results and explain what this reveals about your model's behavior.

There are several parameters which greatly change the outputs even after small manipulation, while others gradually change the output. Most spontaneous ones are listed below:-

- Signal-to-Noise Ratio (SNR)- SNR values directly impacted the output as can be seen in the graphs. As the SNR lowers, At low SNR values (e.g.-20 dB), the received signal is almost buried in noise, making it difficult for both models to distinguish between H_0 (noise) and H_1 (signal present).We can see in the graphs that the value of AUC decreases rapidly as the SNR value lowers.

- Smoothing Factor of Covariance matrix- The smoothing factor determines how many consecutive time samples are taken to compute the covariance matrix.Higher factor means more consecutive time samples which helps the model recognize the pattern over time.This helps GNN detect signals more accurately

- Number of Antennas (M)- More the value of M , larger the Covariance matrix. The GNN explicitly models inter-antenna relationships. So it benefits more from higher M, whereas CNN performance saturates.

- Learning rate and size of Training data- Larger training sets and smaller Learning rates give out better result. Too few samples or too high learning rate causes unstable ROC curves.

9. Discuss any challenges or discrepancies encountered between theoretical expectations and simulation results, and how you plan to address them.

While implementing the SCM-GNN and CNN models for spectrum sensing, we encountered several challenges that created a difference between the proposed theoretical expectations and the simulation results.

- The most difficult task was data generation. Theoretically, we aim to achieve clear covariance matrices that clearly state whether there is a signal or not where signal present is [H1] and the signal absent is [H0]. But in the simulation part, the low SNR values resulted in unstable and noisy covariance matrices. This instability has made learning hard for both CNN and GNN models. We overcame this challenge by adding a smoothing factor equals to 8 while computing the covariance matrices. This helped us to remove the fluctuations and gave more accurate and reliable results.

- There were differences in the theoretical part and in the simulations regarding the number of antennas (M). Theoretically, we know that more antennas should result in better spatial features. Our approach confirms this that when we aimed to increased the GNN accuracy, we significantly increased the M value from 2 to 8, though training time also increased. Increasing the M value, also results in better solutions using CNN. Infact when M=8, both GNN and CNN have exact same AUC values i.e., 1.00. So, when we increase the number of antennas, both CNN and GNN behaves in exact same manner. Therefore, currently we have M=2, where GNN performs better than CNN.

So, we plan to learn the performance of both the methods based on the number of antennas. Also, we plan to integrate the adaptive antenna selection and SpecKriging-based spatial interpolation in our future work. Our current work does not explicitly use SpecKriging but takes motivation from it through correlation-based feature smoothing and graph construction.

10. Provide your team role matrix and screenshots (e.g., from Trello, Slack, Notion, Git). Include a Gantt chart or weekly log to show current progress.

Table 1: Project Meeting Log

Meeting No.	Date	Description
1	12 September, 2025	Had a conversation with Dhaval Sir to determine the quickstart point and how to move forward toward achieving our end goal.
2	10 October, 2025	Had a meeting with the mentor to discuss the paper provided by her and search for reference papers. Also, did a group meeting for discussion.
3	11 October, 2025	Had a meeting with the mentor to finalize the base paper.
4	14 October, 2025	Had the group discussion.
5	15 October, 2025	Had a quick meeting with the mentor regarding the doubts in the base paper.
6	16 October, 2025	Had a final group meeting for project assessment 1.

Table 2: Individual contribution

Name	Enrollment No.	Contribution
Vaishnavi Katba	AU2340048	research, analysis, documentation
Renee Vora	AU2340059	coding implementation, testing, documentation
Sneha Mirani	AU2340071	research, analysis, documentation
Pushti Sonak	AU2340082	research, model formulation, documentation

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.