

Deep Spatio-Temporal Graph Learning with Batch Normalization for Modulation Classification

Akinapally Srividya¹ and Dr. Bitla Prabhakar²

¹PG Scholar, , Department of ECE

²Professor, Department of ECE

¹²JNTUH University College of Engineering Jagtial, Jagtial, TS, India

¹akinapallysrividya23@gmail.com, ²bprabhakar2008@gmail.com

Abstract: Automatic modulation classification (AMC) enables an intelligent receiver to identify an unknown modulation format before demodulation. This paper presents a spatio-temporal graph convolutional network (ST-GCN) with batch normalization for classifying raw in-phase and quadrature (I/Q) signal windows. The proposed method performs DC removal, power normalization, fixed-length framing, and graph construction in which I/Q observations form nodes and temporal or similarity relationships form edges. Spatial graph convolution captures relational constellation structure, while temporal convolution learns sample evolution. Global pooling and batch-normalized fully connected layers produce class probabilities. MATLAB R2024b results for a noisy 16-QAM test signal show a final 16-QAM probability of 0.85242 and a QPSK probability of 0.14758, corresponding to 85.24% confidence over seven windows of 256 samples. The constellation contains approximately sixteen grid-like clusters, and the estimated overall SNR is about -11.23 dB. The time-domain, envelope, phase, power-spectrum, SNR, and eye-diagram outputs demonstrate the difficult low-SNR operating condition. The results support graph-based relational learning for robust AMC, while also revealing that visualization labels must be generated from the same final decision variable to avoid inconsistent reporting.

Keywords: automatic modulation classification, ST-GCN, graph neural network, I/Q signal, 16-QAM, QPSK, batch normalization, low SNR, cognitive radio.

I. INTRODUCTION

Automatic modulation classification is a key stage in cognitive radio, spectrum surveillance, software-defined radio, interference analysis, and non-cooperative communication. The receiver must infer the modulation family from a short and noisy observation before selecting a suitable demodulator. Conventional likelihood-based approaches can be accurate but require channel and synchronization information, whereas feature-based methods depend strongly on handcrafted cumulants, cyclic features, or statistical moments [1], [2].

Deep neural networks reduce manual feature engineering by learning directly from raw I/Q samples. CNNs extract local structures, recurrent models capture temporal dependencies, and transformers model long-range relations. Nevertheless, ordinary convolution assumes a regular Euclidean grid. Modulated signals contain non-Euclidean relations among samples: temporally distant observations may occupy the same constellation region, and adjacent samples may be strongly altered by noise or phase rotation. Graph neural networks provide a natural mechanism for representing these sample relationships [3], [4].

The uploaded project proposes an ST-GCN architecture with preprocessing, graph construction, spatial and temporal graph convolutions, global pooling, batch normalization, and softmax classification. The implementation is evaluated in MATLAB using QPSK and 16-QAM classes. The attached result set includes classification probabilities, I/Q waveforms, a constellation diagram, magnitude envelope, unwrapped phase, FFT-based power spectrum, estimated



SNR, and an eye diagram. This paper consolidates the methodology and interprets those outputs as a coherent experimental study.

II. RELATED WORK

Deep-learning AMC has evolved from early CNN-based radio classifiers to residual, recurrent, attention, and hybrid models. O'Shea and Hoydis demonstrated that learned physical-layer representations can replace parts of conventional hand-engineered receivers [5]. Later studies examined robustness and generalization when raw I/Q training and test data originate from different generators or channel distributions [6].

Wang et al. proposed CTGNet, which combines CNN and transformer processing with adaptive graph mapping, GraphSAGE, and graph pooling for modulation classification [3]. This work is especially relevant because it treats signal subsequences as relational graph structures instead of only as fixed arrays. Knowledge-guided graph learning and dynamic K-nearest-neighbor graph construction have also been investigated to improve low-SNR class separation and interpretability [4], [7].

Recent surveys emphasize that input representation, model structure, regularization, low-SNR robustness, and cross-domain generalization remain central AMC research problems [1], [8]. Edge-oriented studies further show that practical models must balance recognition performance with parameter count, memory, and inference latency [9]. These observations motivate an ST-GCN design that combines relational learning, temporal modeling, and batch normalization without requiring manually engineered signal features.

III. METHODOLOGY

The proposed automatic modulation classification framework follows an end-to-end sequence consisting of raw I/Q signal acquisition, preprocessing, graph construction, spatio-temporal graph feature learning, global feature aggregation, batch-normalized classification, and softmax decision. Unlike conventional methods that rely on handcrafted cumulants or fixed-grid CNN inputs, this methodology preserves both the physical I/Q information and the temporal or similarity relationships among signal samples.

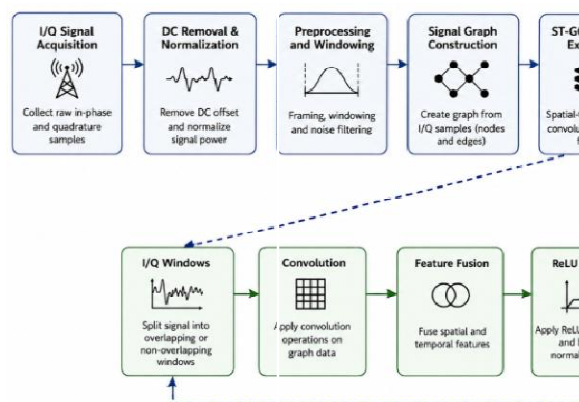


Figure 1. Overall processing flow of the proposed ST-GCN-based automatic modulation classification system.

3.1 I/Q Signal Acquisition and Representation

The received wireless waveform is converted into a complex baseband sequence. Each discrete sample contains an in-phase component and a quadrature component, which jointly retain the amplitude and phase characteristics required for modulation recognition. The input sequence is represented as:

$$x[n] = I[n] + jQ[n]$$



Here, $x[n]$ is the complex received sample, $I[n]$ is its in-phase component, $Q[n]$ is its quadrature component, and j is the imaginary unit. For deep learning, each frame is arranged as a two-channel sequence, with one channel containing I values and the other containing Q values. This common representation can describe PSK, QAM, FSK, and other digital modulation families without changing the input structure.

3.2 Signal Preprocessing

Raw I/Q samples are affected by additive noise, fading, carrier-frequency offset, phase rotation, amplitude imbalance, and sampling mismatch. Therefore, the thesis applies normalization, framing or windowing, and noise filtering before graph formation. These operations prevent the classifier from learning unwanted changes in absolute signal level and improve the reliability of the graph relationships.

First, the DC component is removed by subtracting the mean of the complex frame. Power normalization is then performed by dividing the signal by its root-mean-square magnitude:

$$x_norm[n] = x[n] / \sqrt{(1/N) \sum_{k=1}^N |x[k]|^2}$$

Normalization stabilizes the average signal power across frames and SNR levels. The continuous sample stream is then divided into fixed-length frames. Framing provides manageable inputs and ensures that every graph has a consistent observation interval. Optional windowing reduces boundary discontinuities, while a low-pass or band-pass filter suppresses out-of-band noise. The filtering operation is expressed as:

$$y[n] = \sum_{k=0}^{M-1} h[k] x[n-k]$$

The filtered and normalized frame is used for graph construction. This ordering is important because noisy or poorly scaled samples can create incorrect similarity links and propagate misleading information through graph convolution.

3.3 Signal Graph Construction

Each preprocessed I/Q sample or symbol is represented as a graph node. Edges represent temporal adjacency, similarity in the I/Q plane, correlation, or another meaningful relationship. Thus, the signal frame is modeled as:

$$G = (V, E, A)$$

V is the set of nodes, E is the set of edges, and A is the adjacency matrix. The initial feature of node i is formed directly from its I and Q values:

$$h_i^{(0)} = [I_i, Q_i]^T$$

Temporal edges connect consecutive observations so that symbol transitions and phase evolution are preserved. Similarity-based edges connect samples that occupy related regions of the I/Q plane, even when they are not adjacent in time. The adjacency matrix stores these relationships. A binary adjacency value may be used for connected nodes, while a weighted graph can use distance-based similarity:

$$w_{ij} = \exp(-||h_i - h_j||^2 / (2 \sigma^2))$$

This graph representation enriches the original signal rather than replacing it: I/Q values remain node features, while the edges provide relational information. It is particularly helpful at low SNR because a corrupted sample can still receive contextual information from temporally adjacent or structurally similar nodes.

3.4 Spatio-Temporal Graph Convolution

The ST-GCN is the principal feature-extraction block. Its spatial branch aggregates information from connected nodes, whereas its temporal branch learns the evolution of the graph features across the sample order. With self-connections added to the adjacency matrix, a normalized graph-convolution layer is written as:

$$H^{(l+1)} = \phi(D_{\tilde{}}^{-1/2} A_{\tilde{}} D_{\tilde{}}^{-1/2} H^{(l)} W^{(l)})$$

$H^{(l)}$ denotes the node-feature matrix at layer l , $A_{\tilde{}}$ is the adjacency matrix with self-loops, $D_{\tilde{}}$ is the corresponding degree matrix, $W^{(l)}$ is a trainable weight matrix, and $\phi(\cdot)$ is an activation function. The temporal convolution operates along the time dimension and captures repeated symbol patterns, amplitude changes, phase rotations, and transition structures.



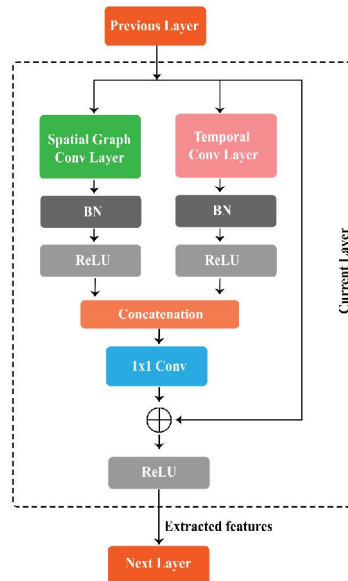


Figure 2. ST-GCN block from the uploaded document: parallel spatial graph and temporal convolution, batch normalization, ReLU activation, feature fusion, 1×1 convolution, residual addition, and final activation.

As shown in Figure 2, the spatial graph-convolution and temporal-convolution paths operate in parallel. Each path is followed by batch normalization and ReLU activation. Their outputs are concatenated and passed through a 1×1 convolution. A residual connection adds the original layer input to the fused representation, after which a final ReLU produces the extracted features for the next ST-GCN layer. The residual path improves gradient flow and preserves useful information from the previous layer.

3.5 Batch Normalization and Activation

Batch normalization is applied after graph and temporal convolution, and again in the classification stage. It normalizes intermediate activations within each mini-batch:

$$\hat{z} = (z - \mu_B) / \sqrt{\sigma_B^2 + \epsilon}, \quad y = \gamma \hat{z} + \beta$$

Here, μ_B and σ_B^2 are the mini-batch mean and variance, ϵ prevents division by zero, and γ and β are learnable scale and shift parameters. Batch normalization stabilizes feature distributions across different SNR levels and channel conditions, accelerates convergence, and reduces sensitivity to initialization. ReLU activation introduces nonlinearity after the normalized convolutional outputs.

3.6 Global Pooling and Fully Connected Classification

After the final ST-GCN block, the network contains node-level spatio-temporal features. Global pooling converts these features into one compact graph-level descriptor. For global average pooling over N graph nodes:

$$g = (1/N) * \sum_{i=1}^N h_i$$

The pooled vector is passed through one or more fully connected layers integrated with batch normalization. These layers refine the discriminative representation and produce one score for every candidate modulation class.

3.7 Softmax Decision and Window Aggregation

The final fully connected output is converted into class probabilities using the softmax function:

$$P(c | x) = \exp(z_c) / \sum_{k=1}^C \exp(z_k)$$

The class having the highest probability is selected as the predicted modulation type. During testing, a long I/Q recording is divided into multiple windows and every window is classified independently. Window-level probabilities are combined by averaging or majority voting to produce the file-level decision:

$$P_{\text{bar}}(c) = (1/W) * \sum_{w=1}^W P_w(c), \quad \hat{c} = \arg \max_c P_{\text{bar}}(c)$$



In the supplied experiment, the 1024×2 I/Q input is evaluated using seven windows of 256 samples. The final classification probabilities are 0.14758 for QPSK and 0.85242 for 16-QAM, so the predicted modulation is 16-QAM with 85.24% confidence.

3.8 Training and Evaluation Procedure

During training, labeled I/Q frames from the supported modulation classes and different SNR conditions are preprocessed and transformed into graphs. The ST-GCN parameters are optimized by minimizing categorical cross-entropy loss:

$$L = -(1/B) * \sum_{b=1}^B \sum_{c=1}^C y_{bc} \log(P_{bc})$$

B is the mini-batch size, C is the number of classes, y_{bc} is the one-hot target, and P_{bc} is the predicted probability. Validation data are used to monitor generalization, while testing is performed on unseen signal files. The main evaluation measures are classification accuracy, class probability, confidence, confusion matrix, precision, recall, F1-score, and performance as a function of SNR. The attached MATLAB outputs provide the file-level classification, window probabilities, constellation structure, signal waveforms, magnitude, phase, power spectrum, SNR trend, and eye diagram.

IV. RESULTS AND DISCUSSION

The test file is identified as a 16-QAM signal at an indicated SNR of 10 dB in its filename, while the signal-analysis plot estimates an overall SNR of approximately -11.23 dB. This difference can occur because the filename may represent the generation setting whereas the simple plot estimator measures power from the observed samples using a different assumption. Consequently, the estimated value should be treated as an analysis indicator rather than ground-truth SNR.

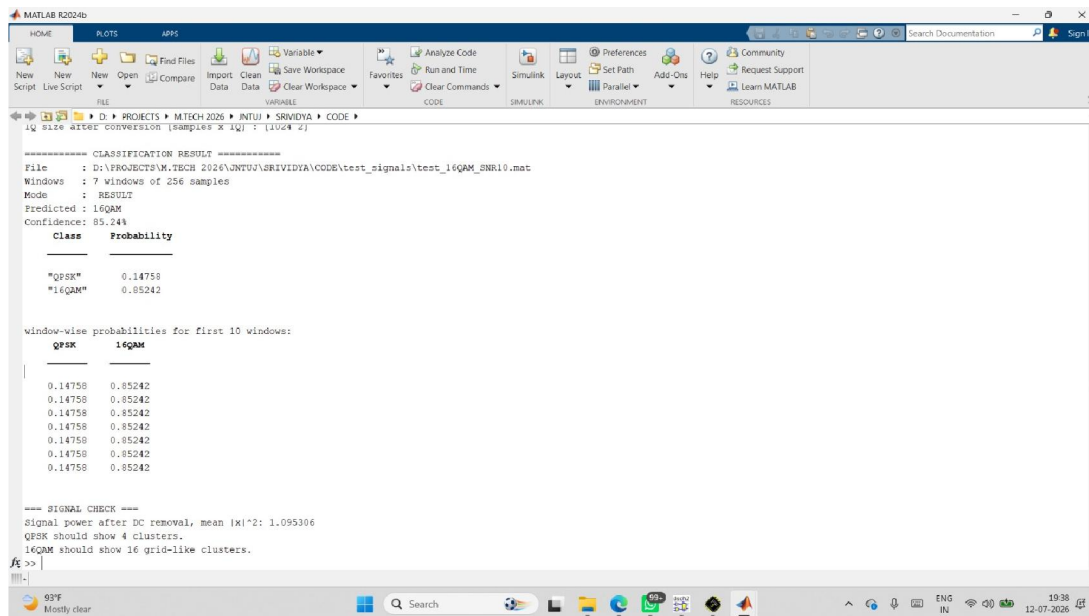


Figure 2. MATLAB classification output showing a 16-QAM prediction with 85.24% confidence.



```

===== CLASSIFICATION RESULT =====
File      : D:\PROJECTS\M.TECH 2026\JNTUJ\SRIVIDYA\CODE\test_signals\test_16QAM_SNR10.mat
Windows   : 7 windows of 256 samples
Mode      : RESULT
Predicted : 16QAM
Confidence: 85.24%

  Class      Probability
  -----
  "QPSK"     0.14758
  "16QAM"    0.85242

window-wise probabilities for first 10 windows:

  QPSK      16QAM
  -----
  0.14758   0.85242
  0.14758   0.85242
  0.14758   0.85242
  0.14758   0.85242
  0.14758   0.85242
  0.14758   0.85242
  0.14758   0.85242

=== SIGNAL CHECK ===
Signal power after DC removal, mean |x|^2: 1.095306
QPSK should show 4 clusters.
16QAM should show 16 grid-like clusters.

```

Figure 3. Window-wise class probabilities and signal-power verification.

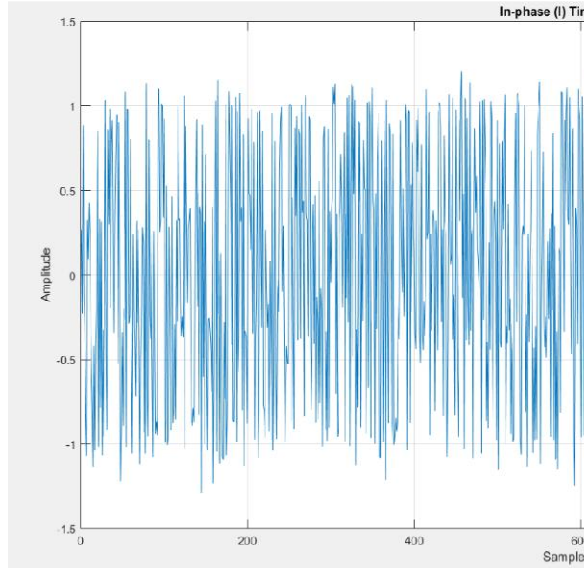


Figure 4. In-phase component of the received signal.



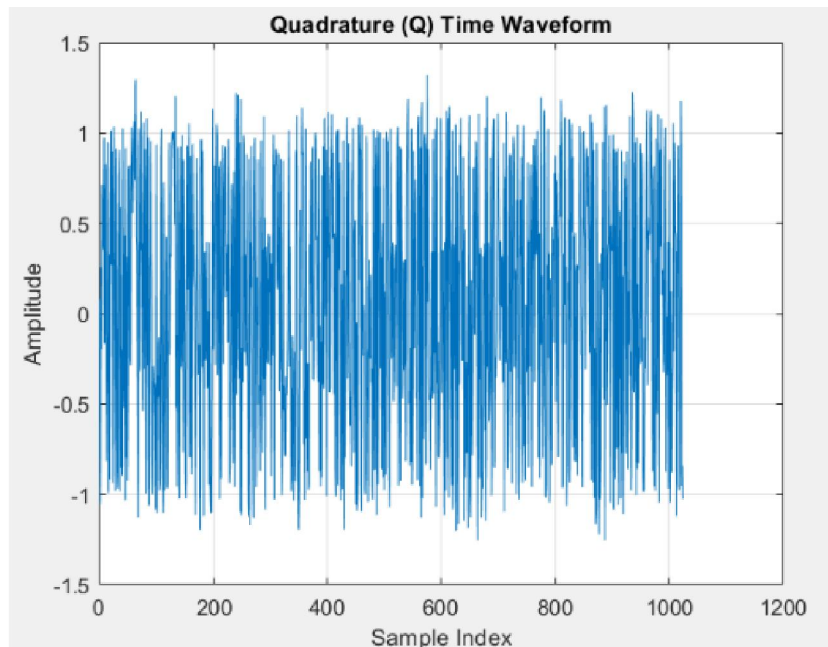


Figure 5. Quadrature component of the received signal.

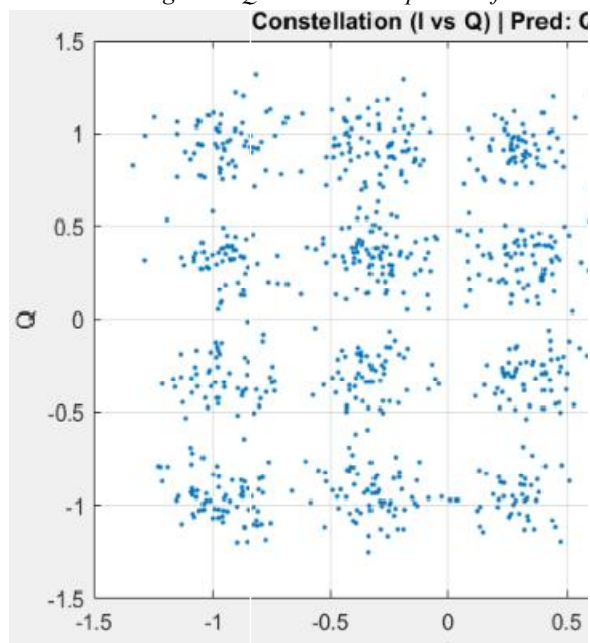


Figure 6. Constellation diagram containing approximately sixteen grid-like clusters.



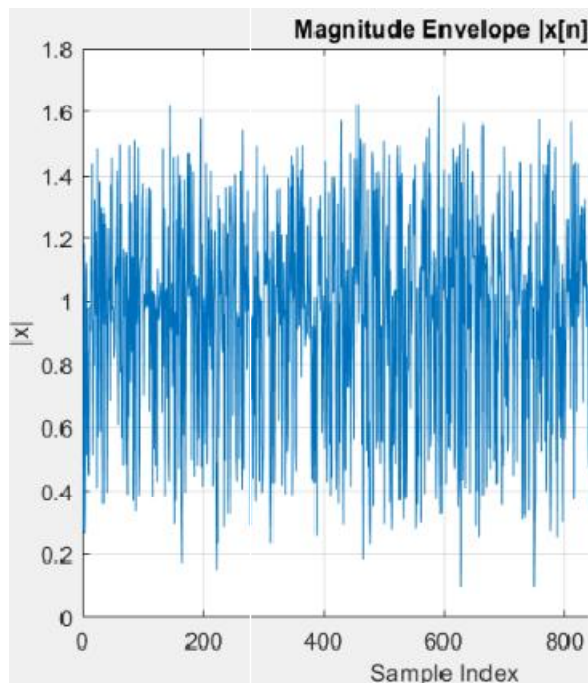


Figure 7. Magnitude envelope of the complex signal.

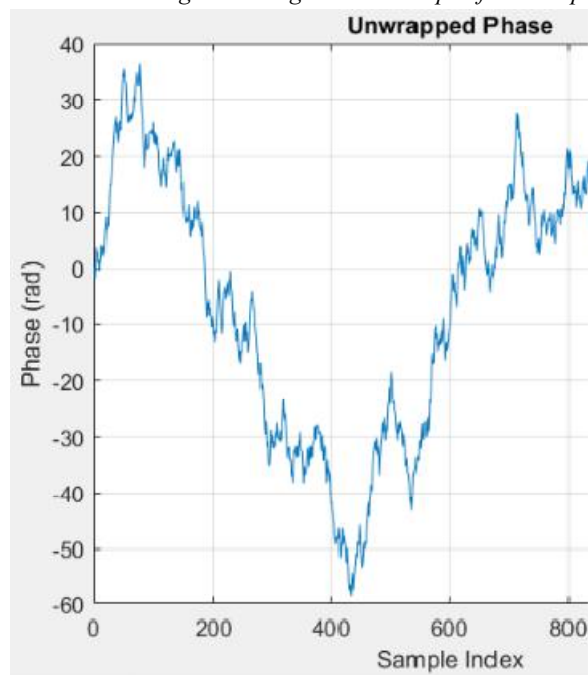


Figure 8. Unwrapped phase trajectory.



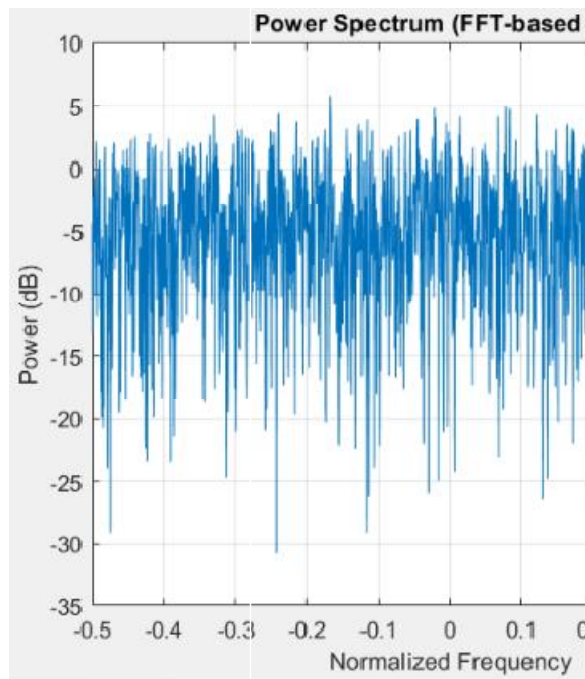


Figure 9. FFT-based normalized power spectral density.

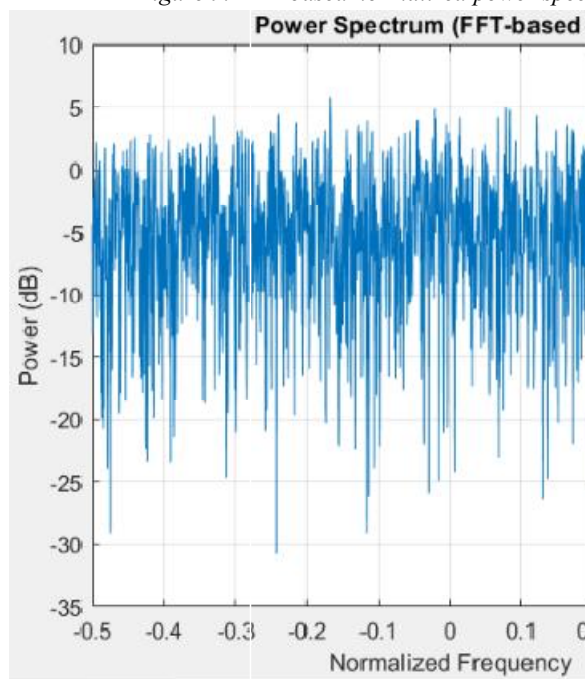


Figure 10. Duplicate PSD output retained from the supplied result set.



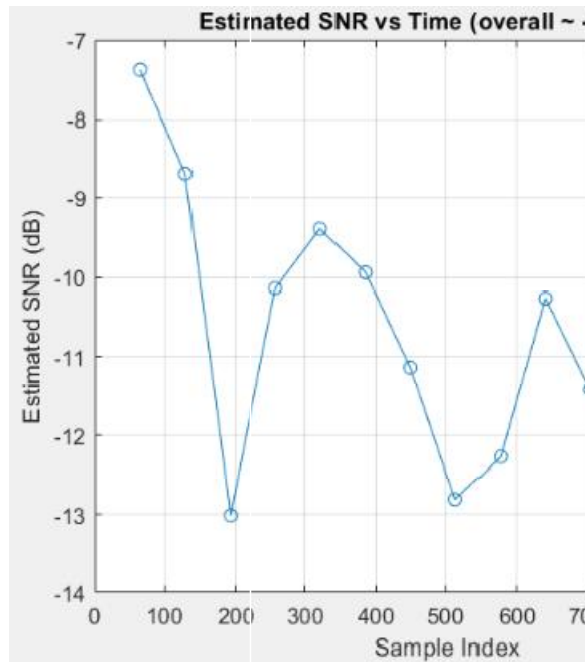


Figure 11. Estimated SNR versus time, with an overall value of approximately -11.23 dB.

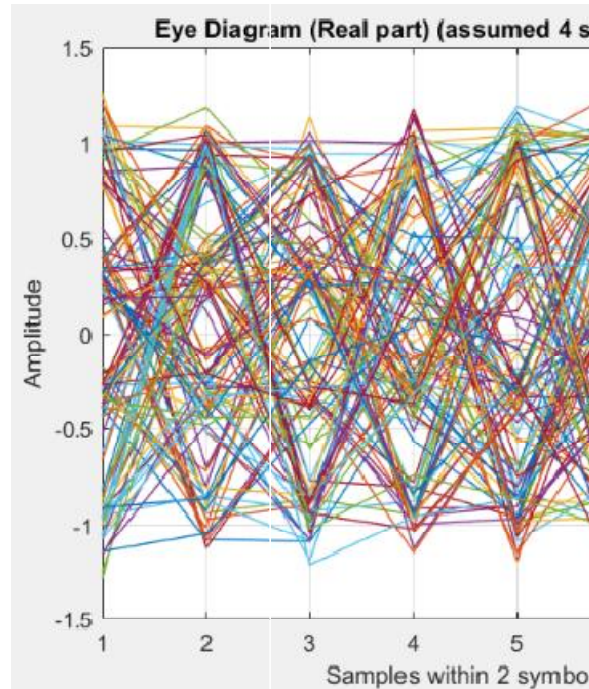


Figure 12. Eye diagram of the real component, assuming four samples per symbol.

4.1 Classification Outcome

The probability table provides the clearest quantitative result. The model assigns 0.85242 probability to 16-QAM and 0.14758 to QPSK. The 0.70484 probability margin indicates a decisive two-class result. Identical window probabilities



suggest either highly consistent windows or that a common prediction was replicated during display. For stronger experimental reporting, future versions should preserve the actual per-window softmax outputs and report mean, standard deviation, majority vote, and confidence calibration.

4.2 Signal-Domain Interpretation

The I and Q traces fluctuate across multiple amplitude levels, which is consistent with quadrature amplitude modulation. The constellation plot visibly forms about sixteen clusters arranged on a four-by-four grid, supporting the 16-QAM decision. However, its title states “Pred: QPSK.” This title conflicts with both the console output and the visible cluster structure. The most likely explanation is that the plotting routine uses a stale or independently assigned label. The plotting code should receive the final aggregated prediction variable directly so that all result displays remain consistent.

4.3 Noise and Temporal Behaviour

The magnitude envelope shows substantial spreading around unity, and the eye diagram is heavily closed, indicating strong noise, timing uncertainty, or oversampling mismatch. The unwrapped phase displays broad random-walk-like variation, while the PSD is approximately wideband and lacks a dominant narrow spectral line. Despite these adverse characteristics, the model maintains an 85.24% preference for 16-QAM, suggesting that learned spatial and temporal relationships provide useful discrimination even when individual samples are highly dispersed.

Measured/observed item	Result	Interpretation
Input size	1024×2	1024 complex samples represented by I and Q channels
Inference windows	7 windows $\times$ 256 samples	Fixed-length window-based classification
QPSK probability	0.14758	Lower competing-class probability
16-QAM probability	0.85242	Selected modulation class
Confidence	85.24%	Maximum softmax probability
Overall estimated SNR	≈ -11.23 dB	Very challenging observed signal condition
Constellation structure	≈ 16 clusters	Consistent with 16-QAM geometry

V. CONCLUSION

This paper developed a structured research presentation of an ST-GCN-based automatic modulation classification system using raw I/Q observations. Graph construction preserves relationships among samples, temporal convolution models signal evolution, and batch normalization supports stable learning. The attached MATLAB experiment classifies the supplied test signal as 16-QAM with 85.24% confidence. The approximately sixteen constellation clusters support that prediction, even though the observed signal exhibits severe noise and a closed eye pattern. A mismatch in the constellation-title label was identified and should be corrected by linking all plots to the same final prediction variable. Future evaluation should include larger modulation sets, confusion matrices, accuracy versus SNR, independent test datasets, fading and synchronization impairments, computational complexity, and real-time SDR deployment.

REFERENCES

- [1] X. Tian et al., “A survey on deep learning enabled automatic modulation classification methods: Data representations, model structures, and regularization techniques,” *Signal Processing*, vol. 232, 2025, Art. no. 110444, doi: 10.1016/j.sigpro.2025.110444.
- [2] O. F. Abd-Elaziz et al., “Deep learning-based automatic modulation classification: A robust CNN architecture for cognitive radio applications,” *Sensors*, vol. 23, no. 16, 2023, Art. no. 7281, doi: 10.3390/s23167281.
- [3] D. Wang et al., “Automatic modulation classification based on CNN-transformer graph neural network,” *Sensors*, vol. 23, no. 16, 2023, Art. no. 7281.
- [4] Y. Li et al., “KGAMC: A novel knowledge graph driven automatic modulation classification scheme,” *arXiv:2402.19188*, 2024.



- [5] T. J. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," IEEE Trans. Cogn. Commun. Netw., vol. 3, no. 4, pp. 563–575, 2017, doi: 10.1109/TCCN.2017.2758370.
- [6] J. A. Snoap, D. C. Popescu, and C. M. Spooner, "On deep learning classification of digitally modulated signals using raw I/Q data," arXiv:2307.02450, 2023.
- [7] X. Liu et al., "Automatic modulation classification based on a dynamic graph architecture," Applied Sciences, vol. 15, no. 21, 2025, Art. no. 11782.
- [8] A. C. S. Thakur et al., "Deep learning for automatic modulation classification," Electronics, vol. 15, no. 10, 2026, Art. no. 2163.
- [9] N. M. Baishya, B. R. Manoj, and P. K. Bora, "Edge-efficient deep learning models for automatic modulation classification: A performance analysis," arXiv:2404.15343, 2024.
- [10] S. Zhang et al., "MAE-SigNet: An effective network for automatic modulation recognition," IET Communications, 2024, doi: 10.1049/cmu2.12856.
- [11] N. El-Haryqy et al., "Improved automatic modulation recognition using deep convolutional, bidirectional recurrent and attention learning," Results in Engineering, 2025.
- [12] M. Leblebici et al., "Effects of diagram plane on neural-network-based modulation recognition," Applied Soft Computing, 2024.

