Capstone Project Presentation

Multi-band Spectrum Prediction Using Graph Convolutional Networks and Recurrent Neural Networks

Ayantan Dandapath
Department of Computer Science,
College of Arts & Sciences
Georgia State University

Georgia State University

Introduction

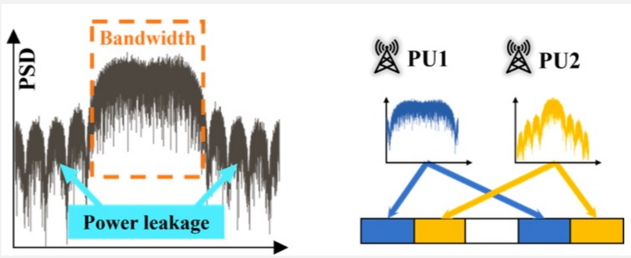
- The proliferation of wireless devices and technologies like 5G, IoT, and smart cities has increased spectrum demand.
- Effective allocation and utilization of limited spectrum resources are critical.
- The need for sophisticated methods to understand and predict spectrum utilization patterns.

Objective

- To explore advanced methods for radio spectrum prediction.
- Focus on three techniques:
  - **GCN + LSTM:** Uses GCNs (Graph Convolutional Network) for spatial structure and LSTM (Long Short-Term Memory) for temporal sequences.
  - **GCN + GRU:** Similar to GCN + LSTM but uses GRUs (Gated Recurrent Unit) for a more computationally efficient alternative.
  - **A-GCRNN:** Combines GCNs and RNNs (Recurrent Neural Networks) with an attention mechanism to enhance prediction accuracy.

Inter-band dependencies and Temporal Dependencies in Spectrum Prediction

- **Inter-band Dependencies:** Relationships and interactions between different frequency bands at a given point in time.

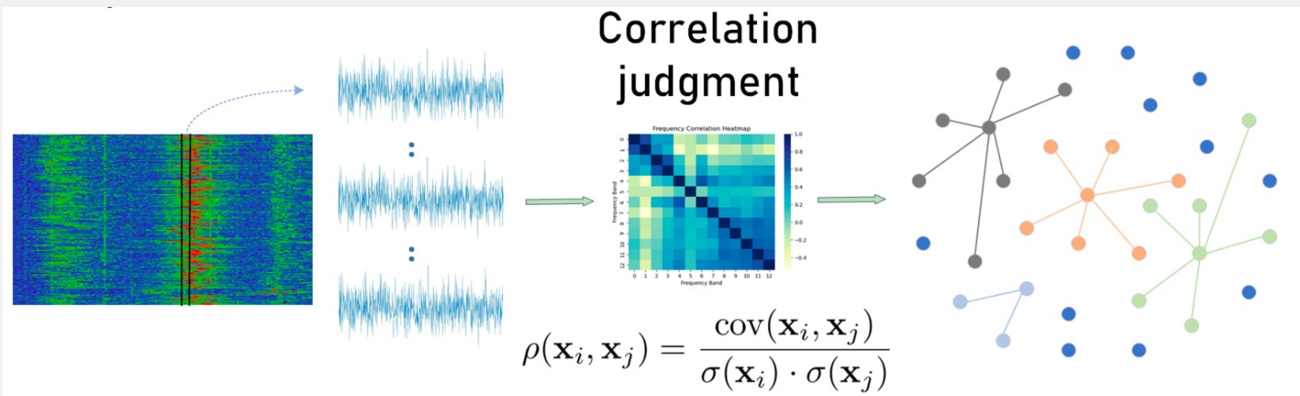


- **Temporal Dependencies**: Relationships and patterns within the same frequency band over different time points.

We're doing it #thestateway

Graph Construction

- Use of graphs to establish relationships between frequency bands (nodes) and their correlations (edges).



where xi, xj represent the PSD (Power Spectral Density) vectors of i and j frequency bands, respectively. P represents the Pearson correlation coefficient, cov(·, ·) represents the covariance function, σ represents the standard deviation.

Problem Formulation

- The goal is to predict future spectrum status based on historical data.
- Graph is described as G = (V,E), where V = $\{v_1, v_2, \cdots, v_N\}$ and N is the number of frequency bands.
- Feature matrices are used to represent the data features of each node and obtain relationships between data. Feature matrix is represented as X, X = $\{x_1, \ldots, x_N\} \in R^{N \times T}$, where T represents the number of spectrum features (the length of the historical spectrum data).
- X_N^t represents the PSD value of each frequency band at time t. Thus, the problem of multi-band spectrum prediction can be considered as learning the frequency band correlations and temporal correlations, and its expression is as follows:

$$\{X_N^{t+1}, \ldots, X_N^{t+l}\} = f(G(V,E), (X_N^{t-n}, \ldots, X_N^{t}))$$

where n is the length of historical spectrum data and l is the length of the spectrum data needed to be predicted.

We're doing it #thestateway

Dataset

- Measurement data is collected from four sensors located in Spain, within a 6 km radius from the center of Madrid is considered. Spectrum situation is created based on the measurement data in a 4 km2 space area of interest.
- The spectrum situation is updated every 1 minute from 1st June 2021 to 8th June 2021.

Dataset	Spectrum measurement data of ElectroSense
Data type	Received signal power in dBm
Location	Madrid, Spain
Number of sensors	3 (test_yago (indoor), test_rpi4 (indoor), rack_2 (outdoor))
Resolution time interval	1 min
Time span	1st Jun. 2021 - 8th Jun. 2021
Frequency span	600 - 700 MHz
Resolution bandwidth	2 MHz

We're doing it #thestateway

GCN + LSTM Model

- Combines GCN for spatial dependencies and LSTM for temporal sequences.
- LSTM Unit:
  - Input, Forget, Candidate and Output Gates:

    $$[i_t, f_t, g_t, o_t] = \sigma(A[x_t, h_{t-1}] W_1 + b_1$$

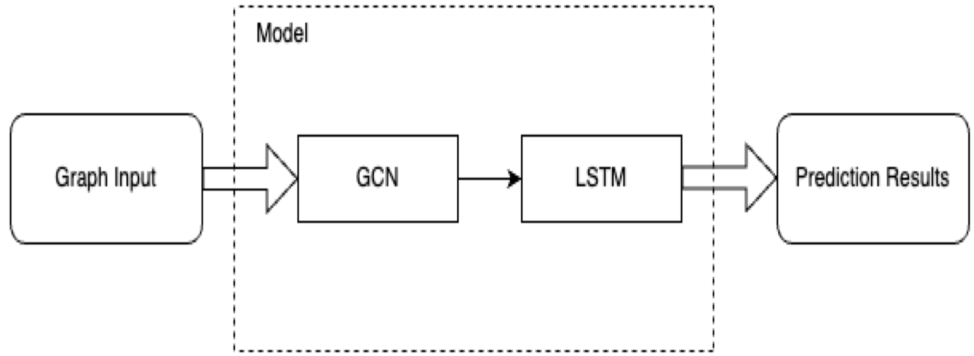
    where A is the Laplacian matrix with self-loops, x_t is the input at time t, h_{t-1} is the hidden state from the previous time step, W_1 and b_1 are learnable parameter and σ is the sigmoid activation function.
  - New Cell State:

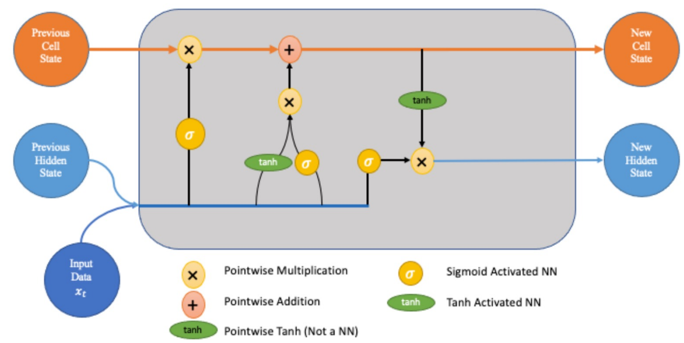
    $$c_t = \tanh(A[x_t, (f_t \odot h_{t-1})] W_2 + b_2)$$

    where f_t is the forget gate, $\odot$ denotes element-wise multiplication, W_2 and b_2 are learnable parameters, and tanh is the hyperbolic tangent activation function.
  - New Hidden State:

    $$h_t = g_t \odot c_t + i_t \odot \tanh(h_{t-1})$$

    where g_t is the candidate gate, i_t is the input gate, and o_t is the output gate.





GCN + GRU Model

- Similar to GCN + LSTM but uses GRU units but GRUs are simpler and more efficient.
- GRU Cell:

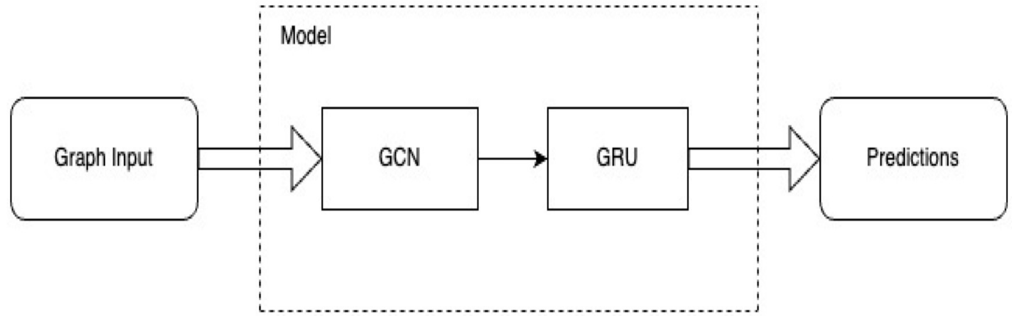
$$z_t = \sigma(W\_z\, X_t^N + U\_z\, h_{t-1} + b\_z),$$

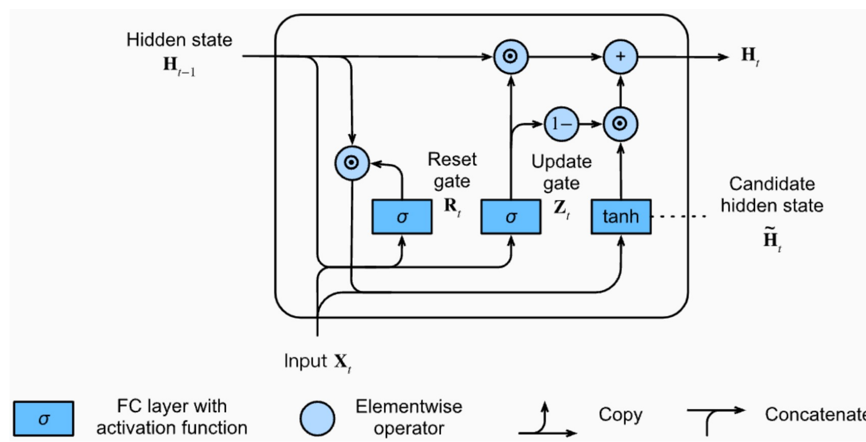
$$r_t = \sigma(W\_r\, X_t^N + U\_r\, h_{t-1} + b\_r),$$

$$\tilde{h}_t = \tanh(W\_h\, X_t^N + U\_h\, (r_t \odot h_{t-1}) + b\_h),$$

$$h_t = (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1}$$

where X_tN represents the PSD value of each frequency band at time t, z_t is the update gate, r_t is the reset gate, h_t-1 is the previous hidden state, $\tilde{h}_t$ is the candidate hidden state, $\odot$ denotes element-wise multiplication, h_t is the current hidden state, and W_z, U_z, b_z, W_r, U_r, b_r, W_h, U_h, b_h are learnable parameters.





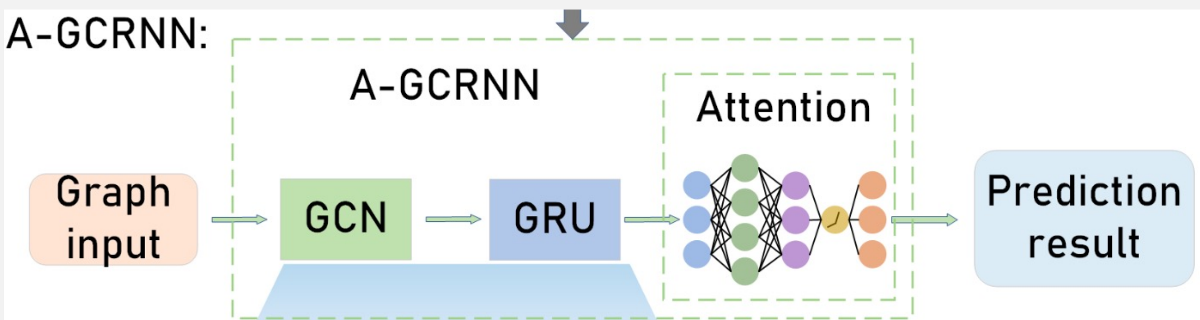
A-GCRNN

- Integrates GCNs, GRUs, and an attention mechanism, leveraging the strengths of each which enhances the ability to focus on important features within the data.
- Attention Module:
  - Soft Attention Network has been employed to compute the weights for the hidden states:

$$\mathbf{e} = \mathbf{W}_a \mathbf{H} + \mathbf{b}_a,$$

$$\widetilde{H} = \sum_{i=1}^{k} \frac{\exp(e_i)}{\sum_{j=1}^{k} \exp(e_j)} h_i,$$

where H is the GRU output hidden state, H = {h_1, h_2, ..., h_k}, k is the number of hidden layers, e is the result of linear weighting, e = {e_1, e_2, ..., e_k }, H is output of the attention network, and W_a and b_a are the learnable parameters of the attention network.



Evaluation Metrics

- Normalized Root Mean Square Error (RMSE)
- Goodness-of-fit performance (R^2)
- Prediction accuracy for PSD values

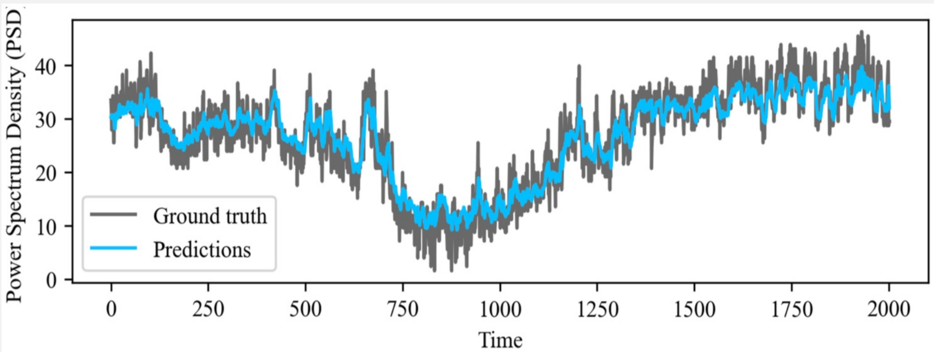
Results

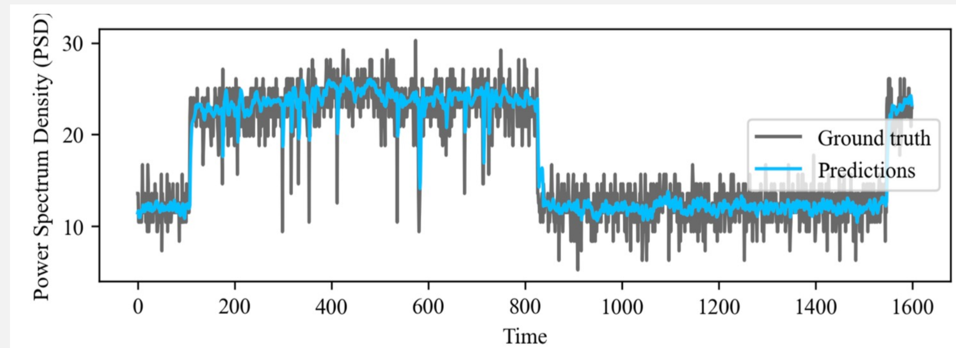
	rack_2		
	RMSE	R2	Accuracy
GCN+LSTM	4.3242	0.67392	0.80165
GCN+GRU	4.6806	0.61798	0.78531
AGCRNN	0.97231	0.87235	0.88924

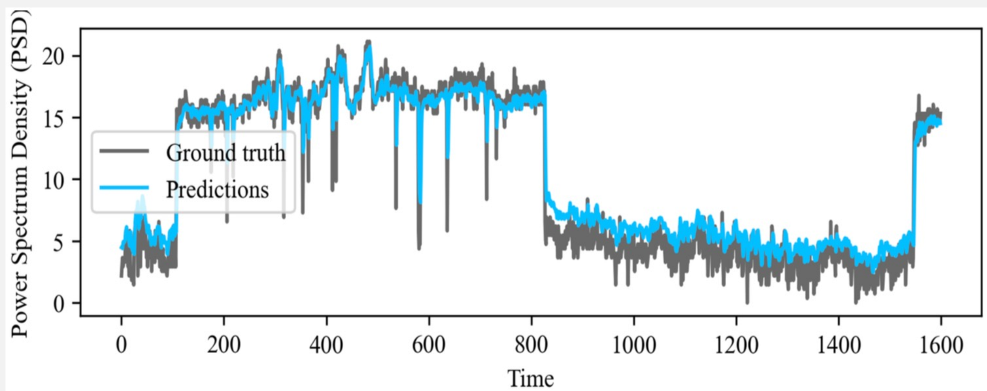
	yago		
	RMSE	R2	Accuracy
GCN+LSTM	3.73630	0.75565	0.83816
GCN+GRU	4.31190	0.67463	0.81323
AGCRNN	0.72751	0.98145	0.93936

	rpi4		
	RMSE	R2	Accuracy
GCN+LSTM	3.11163	0.79688	0.71068
GCN+GRU	3.90086	0.68063	0.73729
AGCRNN	1.21024	0.82709	0.85213

Visualization of Test Set Prediction for rack_2, yago and rpi4 dataset using GCN+LSTM

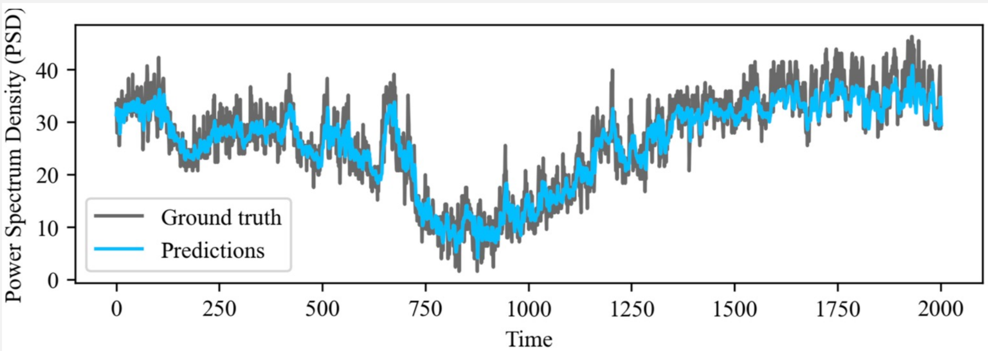


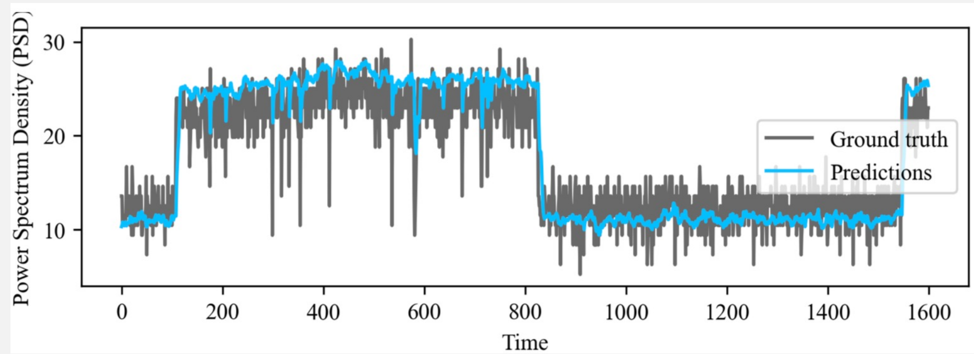


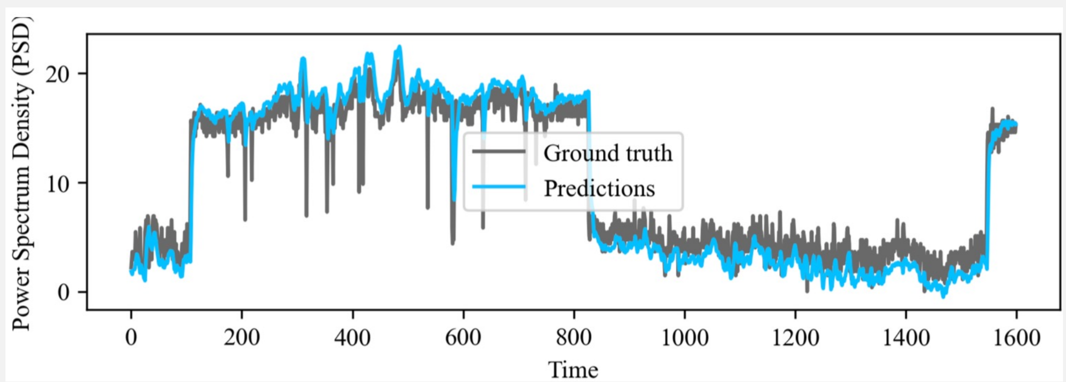


We're doing it #thestateway

Visualization of Test Set Prediction for rack_2, yago and rpi4 dataset using GCN+GRU

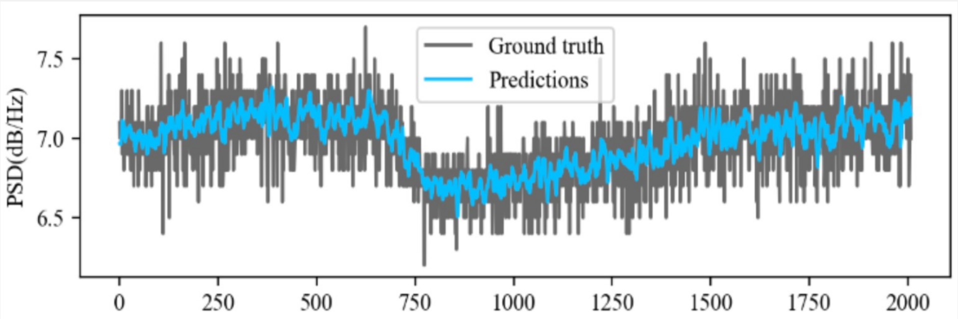


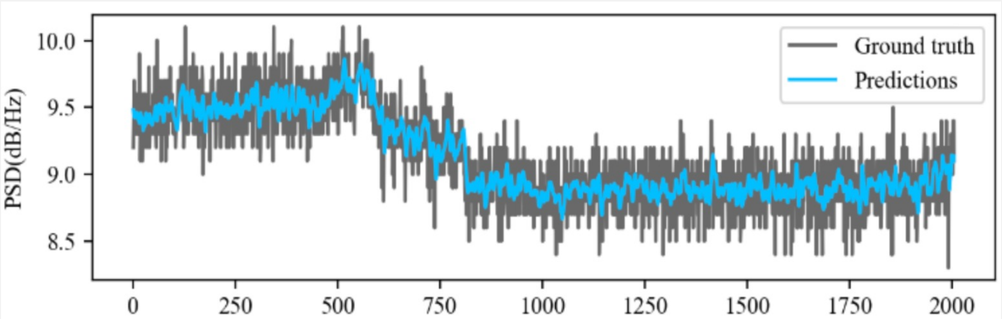


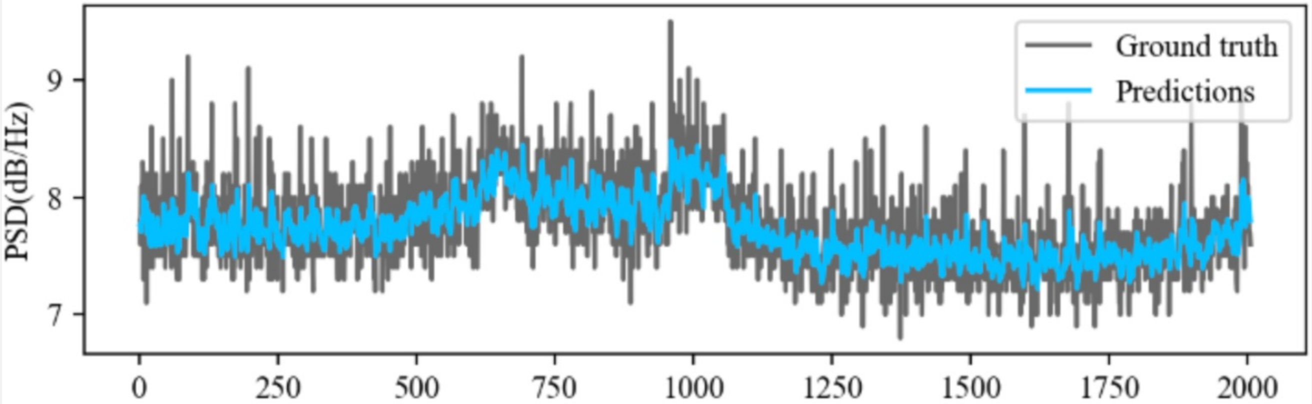


We're doing it #thestateway

Visualization of Test Set Prediction for rack_2, yago and rpi4 dataset using A-GCRNN







We're doing it #thestateway

Conclusion

- GCN + LSTM model effectively captures long-term temporal dependencies and is good at handling complex sequential data. But it struggles with computational efficiency and training time and can sometimes overfit training data, leading to lower generalization performance.
- GCN + GRU model is computationally more efficient and simpler than LSTMs and suitable for scenarios requiring faster computation. But it may not capture long-term dependencies as effectively as LSTMs and is less powerful in modeling complex temporal patterns.
- A-GCRNN model captures spatial and temporal dependencies effectively and specifically the Attention mechanism enhances focus on the most relevant features, improving overall prediction accuracy. But it increase model complexity and computational requirements and may require more careful tuning.