Graph-based EEG Seizure Detection: A Comparative Study

Iris Yazici Hasan Said Unal Semanur Avsar Ender Dogan Isik

I. Introduction

Epilepsy detection, and more broadly the use of electroencephalography (EEG) signals for biomedical tasks, has a long history. Due to their complex structure, EEG signals are challenging to work with, and various signal processing methods have been applied in the past to address this. With the advancements of machine learning (ML), learning-based methods have been successfully applied to biomedical signals. A growing direction in the ML literature involves graph-based learning methods, which operate on graphs. Applying graph-based techniques or introducing graph structures over complex data has become increasingly popular. The representational power of graphs makes them well-suited for a wide range of tasks and can potentially lead to improved performance over traditional methods.

In this project, our goal is to model multi-channel EEG signals as time-dependent graphs and apply graph learning methods to predict the presence of a seizure within a given time window. Our challenges include preprocessing the EEG signals before applying ML methods which is a general issue with EEG signals, constructing a graph structure where each node represents a channel and edges that highly represents the data, and training models that achieve performance better than traditional machine learning methods. We describe our approach in the Methods and discuss our findings and interpretations in the Results section.

II. Methods

A. Preprocessing Methods

To extract meaningful representations from raw EEG signals, we experimented with different signal processing techniques. These features serve as the input node attributes for our graph-based classification models.

- **Fast Fourier Transform (FFT)**: Obtains the frequency content of the signals by transforming them into the spectral domain. The log FFT magnitudes were used with a Band Pass filter (between 0.5 and 30 Hz).
- **Short-Time Fourier Transform (STFT)**: Divides the signal into overlapping windows and applies FFT to each segment. This allows us to observe how frequency content evolves over time which is useful for identifying dynamic events such as seizures.
- **Discrete Wavelet Transform**: For each EEG channel, this transformation decomposes the signal into multiple frequency components where we selected the coefficients from levels D2 to D5.
- **Bandpower (via Welch's method)**: Estimates the signal power within standard EEG frequency bands (e.g., delta, theta, alpha, beta).
- **Combined Feature Representation**: For each EEG channel, we concatenated the wavelet, bandpower, and STFT features into a single vector.

B. Graph Construction Methods

To represent the relationships between EEG electrodes as graphs, we explored different graph construction strategies, where each electrode represents a node.

- **Unweighted graph structure according to the standard 10-20 montage for EEG acquisition**: This representation is based on distances between electrodes on the scalp, but it is unweighted.
- **Inverse Distance-Based Graph**: We used the 3D coordinates of the electrodes to compute pairwise distances. Then, we created edges between all pairs of electrodes, assigning edge weights as the inverse of the Euclidean distance. This encourages stronger connections between physically closer electrodes. The resulting graph is undirected and fully connected, excluding self-loops.
- **k-Nearest Neighbor (k-NN) Graph**: In this approach, each electrode (node) is connected to its k closest neighbors in 3D space. This yields a sparser and more localized graph than the fully connected distance-based version. All connections are treated as undirected and unweighted.
- **Mutual Information-Based Graph**: We calculated pairwise mutual information (MI) between EEG channels, capturing statistical dependencies in the signal instead of just spatial proximity. We discretized the signals, computed normalized MI between all channel pairs across many EEG samples, and built an undirected graph by connecting each channel to its top-k MI neighbors (e.g., k=19, fully connected). The edge weights correspond to the MI values.
- **Correlation Based Graph**: Based on the work in [1] we calculated normalized cross-correlation between the EEG channels, and constructed the graph by connecting each channel to its top-k neighbors, and edge weights corresponds to cross-correlation values.

C. Models

To classify EEG recordings as seizure or non-seizure, we constructed graphs where nodes represent electrodes and edges capture spatial or functional relationships built from the graph

construction methods section. We then applied various graph based learning algorithms to learn the classification. Each model extracts local and global features using message-passing mechanisms, followed by pooling to produce a graph-level representation for classification. Below is a description of the models explored:

GCN (Graph Convolutional Network): The GCN performs spectral convolution by approximating the graph Laplacian with a first-order neighborhood filter. It aggregates feature information from immediate neighbors, weighted by the graph structure. We used an architecture with two GCN layers followed by global mean pooling and a classification head with also batch normalization and dropout for better generalization and regularization. We also experimented with concatenating mean and max pooling.

ST-GNN (Spatiotemporal Graph Neural Network): The temporal encoder (either LSTM or 1D CNN) acts as a feature extractor, learning embeddings related to time-series structure for each EEG electrode [2]. These features are then passed on to GCN layers. Finally, the spatial node representations are mean pooled to get a graph-level representation.

GraphSAGE (SAmple and aggreGatE): GraphSAGE learns node embeddings by aggregating features from a node's neighborhood using a learnable aggregation function. It increases scalability by subsampling the neighborhood [5]. But it does not use the edge weight information making some of our graph construction methods dysfunctional. We used an architecture with two SAGEConv layers followed by global mean pooling and a classification head.

ChebNet (Chebyshev Graph Convolution): ChebNet is a spectral method for learning on graphs that defines convolutional filters as a truncated expansion of Chebyshev polynomials of the graph Laplacian [3]. Graph Convolutional Network (GCN) can be interpreted as a simplification of ChebNet keeping only the first-order term (K=1) in the Chebyshev polynomial. [4] We implemented three versions:

- EEG_ChebNet: Two ChebConv layers followed by global mean pooling and a classification head with also batch normalization and dropout.
- EEG_ChebNet_Attn: Extended version with a global attention mechanism for pooling, where a learnable gate determines the importance of each node's embedding.
- EEG_ChebNet_Res: Extended version having residual connections after each convolution to help preserve gradient signals and preventing over-smoothing.

Graph Isomorphism Network (GIN): The main intuition of GIN is to maximize the representation power of GNNs by using injective aggregation function [6]. We used an architecture with two GIN layers, that also incorporate edge features into the aggregation function, followed by concatenating add, mean and max pooling and a classification head with dropout. We also added random dropout of edges in training mode.

GAT (Graph Attention): GAT performs attention operation for each layer. Each node learns the attention weights of its neighbors that decides the importance of each neighbor's aggregated message to its updated embedding [7]. Since it learns the attention weights, we did not use the edge weights in the GAT setup and left that part to the model. We used multihead attention and concatenate them in all of our trials. We experimented with different variations and here we present two of them:

- EEG_GAT: the model applies global mean pooling to achieve a single vector embedding for the whole graph and makes predictions based on this embedding.
- EEG_GAT_superpool: we add a super-node that is connected to the whole graph. We used masked attention so that the supernode is updated by the whole graph whereas it does not aggregate its embedding to other nodes. This way, we aimed to the model to learn which nodes are more valuable in its predictions by learning attention weights while not adding extra heads or less informative pooling methods.

D. Training

To train our models, we implemented a standard supervised training loop in PyTorch. The dataset was first split into 20% training and 80% validation set using stratified sampling to preserve the imbalanced class distribution (20% seizure). When splitting the data by patient IDs, we found inconsistencies (97 unique IDs vs. reported 50 patients) and observed mixed seizure/non-seizure segments even within sessions. Due to these, we mostly used segment-level splitting instead, though this may inflate validation performance for new patients. We also tried 3-fold cross-validation. We optimized our models over a weighted binary cross-entropy loss (BCEWithLogitsLoss) with a pos_weight parameter derived from the ratio of class instances. We used the Adam optimizer with a learning rate scheduler (ReduceLROnPlateau) that adaptively reduced the learning rate based on validation performance.

During training, we tracked both loss and macro-averaged F1 score on the training and validation sets at each epoch. The macro F1 score was chosen as the main metric due to the class imbalance and the importance of capturing both sensitivity and specificity.

At the end of each epoch, the model's performance on the validation set was evaluated without gradient updates. The model achieving the highest validation F1 was saved for final testing. We have also employed early stopping with different patience values.

III. Results

We have experimented with different combinations of preprocessing, GNN architectures and graph construction techniques described in the Methods.

A. GCN

1) Comparison of different graph construction methods on GCN: GCN uses edge weights during training hence we have tested the inverse distance based and mutual information based graph construction methods which result in weighted graphs. Validation F1 score and loss comparison can be seen in Fig. 1. We have observed that MI based construction provides better

and more stable performance than using just spatial distance, likely because MI captures signal dependencies beyond physical proximity.

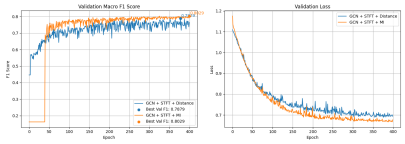

Fig. 1. F1 and Loss comparison across different graph construction methods using GCN.

2) Comparison of different preprocessing methods on GCN: We have compared different preprocessing methods. Validation F1 score and loss comparison can be seen in Fig. 2. We have observed that, Wavelet and Bandpower perform poorly in terms of validation performance. Wavelet coefficients capture high-frequency localized events, but seizure patterns in EEG may also involve slower, more distributed changes. Bandpower, on the other hand, extracts only four features per channel by summarizing power over broad frequency bands. While interpretable, this representation may be too simple to capture the subtle and fast-changing patterns typical of seizure activity. In contrast, FFT offers results close to STFT but with more fluctuation during training. STFT achieves the most stable and highest-performing results, thanks to its fine-grained time-frequency resolution. The Combined approach, which concatenates STFT, Wavelet, and Bandpower features, achieves the highest F1 score overall but suffers from increased fluctuation. This instability is likely due to the limitations introduced by the less effective components (Wavelet and Bandpower), which may inject redundant or noisy features. In addition, as it can be seen in Fig. 2, the positively weighted BCE loss might not always align with the macro-F1 score.

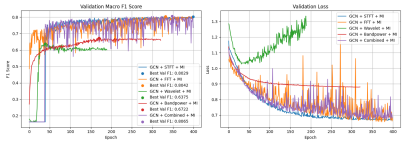

Fig. 2. F1 and Loss comparison across different preprocessing methods using GCN.

The comparison above is for when the graph construction is based on MI. For distance based graphs, we have observed that Wavelet features outperform FFT features, unlike MI graphs. The macro-F1 scores for FFT and Wavelet are given in Table I while Fig. 3 shows the loss and F1 score for Wavelet transform. This implies that different pre-processing strategies might work better with different graph structures.

B. ChebNet

We have tested the three different ChebNet variants we mentioned. Validation F1 score and loss comparison can be seen in Fig. 4. We have observed that residual connections in ChebNet significantly improve model performance by easing

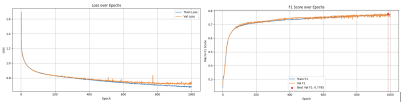

Fig. 3. Wavelet pre-processing with a distance based graph

optimization and stabilizing training. Attention pooling also improves learning but is slightly less stable than residuals alone. Overall, ChebNETRes + STFT + MI is the most effective architecture among the three. Overall using ChebNET increased the performance compared to the GCN. This can be due to the fact that ChebNET is actually a broader version of GCN as explained in the methods section.

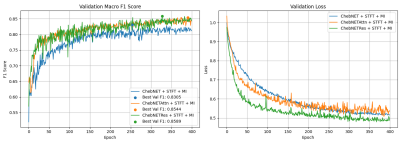

Fig. 4. F1 and Loss comparison across ChebNet variants.

C. ST-GNN

ST-GNN uses 3000 raw time points, hence no preprocessing. Therefore, the input features are high-dimensional compared to preprocessing techniques like bandpass filtered FFT coefficients. The CNN layer or the LSTM first extracts features using the time-series nature of the signal, which is followed by GCN layers forming a spatial representation. We have tried both CNN and LTSM as the temporal encoder. However, we observed that the training time for LSTM was much higher and the maximum validation macro F1 score did not increase beyond 0.7. Therefore, we opted for the simpler CNN layer, and used a distance graph. Fig. 5 shows that the validation F1 score is 0.78 and the validation loss drops with training loss, without overfitting.

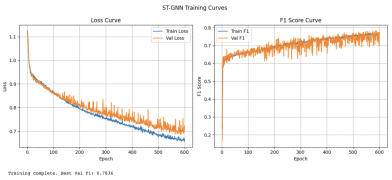

Fig. 5. ST-GNN: Loss and macro-F1 score for train and validation sets

D. GraphSAGE

GraphSAGE does not use edge weights during training hence we have tested the fully connected and k-NN based graph construction, experimenting with different k values. Validation F1 score and loss comparison can be seen in Fig. 6.

Among the graph construction methods, using a k-NN graph with k = 5 achieves the highest best validation F1 score slightly outperforming the k=10. The fully connected version

performs the worst. Hence, increasing connectivity decreases performance. Increasing connectivity may introduce noisy or redundant edges, diluting the signal from relevant neighbors and making it harder for the model to focus on the most meaningful relationships. Furthermore, the results are overall very noisy and the performance fall below the GCN. This suggests that GCNs may better exploit the full graph structure in this context, whereas the sub-sampling in GraphSAGE might discard important connectivity patterns.

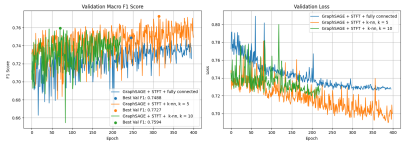

Fig. 6. F1 and Loss comparison of GraphSAGE for different graph construction methods.

E. GIN

When experimenting with GIN, we first split the dataset at the segment level, the learning curves was performing well, but test score has experienced a drop. So, we experimented by splitting based on patient ids. Our best configuration came from combined transform in preprocessing, and correlation based graph construction, and also we observed that random dropout of edges helps generalization. Validation F1 score and loss comparison can be seen in Fig. 7. With this setup, the public leaderboard score closely matched our validation score.

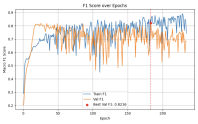

Fig. 7. F1 evolution of GIN.

F. GAT

We have experimented with two GAT variants we mentioned before. Since the attention weights are learned during training, we used unweighted graph structure but rather focused more on pre-processing signals. Our best results come from the normalized combined features from signals. We used attention masking and dropout to prevent overfitting. Among two, GAT with supernode generalized better. Since the supernode learns the importance of all nodes in prediction as a graph element, it is possible to say that instead of global pooling, adding a supernode for readout fits more naturally to the task. This would be the main reason why GAT with supernode generalizes better than default GAT, even if they have similar F1 scores over train and validation test.

G. Overall Comparison of Methods

We compare our models based on the validation macro-F1 score, shown in Table I. The confusion matrix for our best model is shown in Fig. 8.

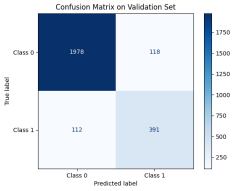

Fig. 8. Confusion matrix of the best model on the validation set

TABLE I
COMPARISON OF METHODS

Method	Validation Macro-F1 Score
GCN + FFT + Unweighted	0.6867
GCN + FFT + Distance	0.7113
GCN + STFT + Distance	0.7879
GCN + STFT + MI	0.8029
GCN + FFT + MI	0.8042
GCN + Wavelet + MI	0.6375
GCN + Wavelet + Distance	0.7785
GCN + Bandpower + MI	0.6722
GCN + Combined + MI	0.8065
ST-GNN + Raw data + Distance	0.7836
ChebNET + STFT + MI	0.8305
ChebNETAttn + STFT + MI	0.8544
GAT + Combined + Unweighted	0.8416
GATSuper + Combined + Unweighted	0.8401
ChebNETRes + STFT + MI	**0.8589**

IV. CONCLUSION

GCN and ChebNet models achieved higher performance and showed more stable optimization during training, compared to GIN and GraphSAGE. The ChebNET models and GCN+Combined+MI achieved F1 scores greater than 0.8 on the validation set and also on the public leaderboard on Kaggle, but their performance dropped significantly on the private leaderboard. The varying performance of these models suggest that they do not generalize well for different subsets of data, which might have different class distributions. Even though we tested our model with k-fold cross-validation and different random and straified train-test splits and we also tried splitting according to patient ids, most of our models had high performance on both validation set and public leaderboard, but low performance in the private board while the wavelet preprocessing combined with distance graph suffered from having high performance in validation and private leaderboard, but low performance on the public leaderboard. These issues are likely due to overfitting. However, one of our models, which used FFT with two GCN layers and a distance based graph, achieved stable performance with a macro-F1 score well above 0.7 in both public and private leaderboards, and the validation set. This was one of our simplest models in terms of the preprocessing and architecture.

Overall, in this work, we observed the benefits of using graph-based methods on increasing the performance in EEG seizure detection. We achieved F1 scores up to 0.86, which outperformed non-graph approaches such as the LSTM. We conclude that the relationships between electrodes really matter in seizure detection.

REFERENCES

[1] Tang, S., Dunnmon, J., Saab, K., Zhang, X., Huang, Q., Dubost, F., Rubin, D. & Lee-Messer, C. Self-Supervised Graph Neural Networks for Improved Electroencephalographic Seizure Analysis. (2022), https://arxiv.org/abs/2104.08336

[2] Al Sahili, Z., & Awad, M. (2023). Spatio-Temporal Graph Neural Networks: A Survey. arXiv preprint arXiv:2301.10569. [Online]. Available: https://arxiv.org/pdf/2301.10569

[3] Defferrard, M., Bresson, X., & Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. Advances in Neural Information Processing Systems (NeurIPS), 2016. [Online]. Available: https://arxiv.org/pdf/1606.09375

[4] Kipf, T.N., & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. International Conference on Learning Representations (ICLR), 2017. [Online]. Available: https://arxiv.org/pdf/1609.02907

[5] Hamilton, W.L., Ying, R., & Leskovec, J. Inductive Representation Learning on Large Graphs. Advances in Neural Information Processing Systems (NeurIPS), 2017. [Online]. Available: https://arxiv.org/pdf/1706.02216

[6] Xu, K., Hu, W., Leskovec, J. & Jegelka, S. How Powerful are Graph Neural Networks?. (2019), https://arxiv.org/abs/1810.00826

[7] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. Graph Attention Networks. arXiv preprint, 2018. [Online]. Available: https://arxiv.org/abs/1710.10903