DGL-24 Kaggle Competition: GrafT-Heory

Yasmine Hemmati *CID: 02543699*
Alex Iordachescu *CID: 02524916*
Hugo Frelin *CID: 01721441*
Igor Sadalski *CID: 01848914*
Alan Picucci *CID: 02482490*

I. Introduction

In recent years, the analysis of high-resolution brain connectivity graphs has become crucial in detecting and diagnosing various neurological disorders. Recent technological advancements in the medical imaging field enabled significant increases in spatial resolution, which facilitates the representation of fine structural details [1]. Despite these advancements, the acquisition process remains hindered by several factors, such as time, hardware and deterioration in signal-to-noise ratio [2]. The scarcity of high-resolution medical data hampers the progress in neurological disorder diagnosis and treatment. By harnessing recent developments in deep learning and graph neural networks, our work proposes a solution that can efficiently increase the resolution of readily available lower-resolution graphs, thereby making high-resolution brain connectivity analysis more accessible and cost-effective.

In this report, we investigate how to convert low-resolution connectivity graphs to high-resolution ones using generative Graph Neural Networks (GNNs) trained in an inductive setting. Brain graph super-resolution remains a challenging problem due to the non-Euclidean nature of graph data. Current state-of-the-art solutions, such as unsupervised multi-topology approaches [3], have shown promising results in addressing this challenge. We build on previous work [4] and propose SuperBLTGraph, a novel architecture showing promising results on the competition dataset.

II. Dataset

The three-dimensional structure of the human brain can be abstracted into graph form, where nodes represent different regions of the brain and edges represent neural connections between these regions. We can express these graphs as adjacency matrices, where each element represents the strength of the connection between nodes. Our dataset thus contains the adjacency matrices of 279 pairs of low- and high-resolution graphs, having 160 and 268 nodes respectively. Notably, we only use 167 such samples for training and validation. Several pre-processing steps are applied to the raw adjacency matrices to allow for efficient analysis. Firstly, the adjacency matrices are stored as one-dimensional vectors obtained by stacking the columns in the upper-triangular matrix, exploiting the symmetry conferred by undirected graphs to allow for a more compact representation. Secondly, negative and "NaN" values are replaced with 0's. Therefore, we apply similar post-processing steps to our model's outputs to ensure consistency in data formatting and distribution.

III. Methods

A. Base Model

We use the Graph Super Resolution [4] architecture as our base model, comprised of a Graph U-Autoencoder, a Graph Super-Resolution layer (GSR), and a stack of Graph Convolutional (GCN) layers.

The U-Autoencoder is based on the U-Net encoder-decoder architecture [5] and aims to generate meaningful node feature embeddings by encoding the graph topology and decoding that information. Initially, the nodes are attributed identity embeddings, which are passed to the U-Autoencoder along with the connectivity matrix. The Graph Super-Resolution layer uses these embeddings along with the low-dimensional connectivity matrix to generate high-dimensional representations of both the connectivity matrix and the node embeddings. Finally, the stack of GCN layers propagates the topological relationships between the nodes in the high-dimensional space to update the node representations.

The loss function uses the Mean Squared Error (MSE) L_{hr} between the super-resolved connectivity matrix and the ground truth. However, this is further regularised by supervising the intermediate results in the pipeline, via the self-reconstruction loss L_{rec} and the eigen-decomposition loss L_{eig}. The L_{rec} consists of the MSE between the initial node embeddings and the outputs of the U-Autoencoder and has a tunable contribution λ, while the L_{eig} is represented by the MSE between the weights generated by the GSR layer and the eigenvectors of the ground truth. Thus, the base model's loss is given by: $L = L_{hr} + L_{eig} + \lambda L_{rec}$.

B. Embedding Generation

Images can be viewed as instances of graphs, where the nodes are positioned on structured 2D grids. The initial architecture utilised graph U-Nets to upsample the low-resolution brain graph. Graph U-Net incorporates graph pooling and unpooling operations to downsample and upsample the graph, respectively, preserving both local and global structural information. It also integrates Graph Convolutional layers within its blocks to extract features from the input graph efficiently.

We decided to experiment with removing the Graph U-Net and replacing it with a GCN to upsample the low-resolution images. This is motivated by the considerable increase in the number of operations that the former brings, which will be inevitably reflected in the training time and resources used. We therefore aim to achieve good results with a simpler model. Thus, we argue that a GCN may be sufficient to learn good

representations, as a single information diffusion step offers plenty of information about the graph topology.

Consequently, we decided to remove the associated self-reconstruction loss, which was designed specifically for the U-Net. While it could have still been adapted to our model, doing so would not have been justifiable, given the shallow learning scenario it would have been concerned with.

C. Data Distribution shifting

Due to the pre-processing performed on the data, the distribution of the connectivity matrix weight has an abnormally large spike at 0. In order to facilitate the learning of this distribution, we combined a series of architectural and data-processing tricks. Firstly, we designed our model such that the final activation used is Tanh, which allowed it to predict adjacency matrix weights between -1 and 1. We then shifted all zero values in the training target data to -0.05. This creates a greater discrepancy between the edges that are non-zero and the edges that are zero, and thus a larger penalty for incorrectly setting the edge to a value greater than 0. After the loss is computed, we then clip the negative values to back zero before computing the validation MAE. The outputs are also clipped during inference. The goal of these methods is to enhance the discrepancy between zero-valued and non-zero-valued edges, such that the model more accurately can learn to reproduce the target distribution. The results are shown in Figure 3.

D. Objective criterion

We propose the use of another criterion in the loss function, namely the Mean Absolute Error (MAE) instead of the Mean Squared Error (MSE). Typically, MAE encourages feature sparsity during training, determining the model to only learn the most important features. In a graph learning setting, prioritising features may result in a higher focus on the macroscale topology. While MSE is typically the default loss function for regression problems, at times MAE has been shown to perform better in similar scenarios such as image restoration, e.g. due to a higher ability to escape suboptimal local minima [6].

E. Hyper Parameter Tuning

We optimized our model using a Bayesian hyper-parameter search, preferring this approach to a standard random or grid search. This method effectively navigates the hyper-parameter search space by learning from previous results which parameter values lead to optimal performance. The performance of each configuration was determined through the best MAE obtained on a held-out validation set accounting for 25% of the full training set. We set the number of training epochs to 150, but in order to avoid excessive training times and overfitting, we implemented an early stopping mechanism. Moreover, to help the model escape local minima, we implemented a scheduler which halves the model's learning rate after the validation MAE has plateaued for 5 epochs. The hyperparameter space we searched over is: learning rate (1e-4, 2e-4, 3e-4, 4e-4, 6e-4, 8e-4), hidden dimension (320, 640, 960, 1280, 1600), dropout rate (0, 0.1, 0.2, 0.3, 0.4, 0.5), weight decay (0, 0.01, 0.001, 0.0001). Moreover, the option of using two GCN layers before the GSR layer instead of just one layer was used as a hyperparameter, and ultimately was implemented in the final model as it seemed to perform better.

F. Permutation invariance and equivariance properties

The primary building blocks of SuperBLTGraph are the GCN layers, which aggregate node features based on their weighted connection: this operation is permutation-invariant. Moreover, the matrix operations in the GSR layer are also inherently permutation-invariant. This yields permutation invariance through all of SuperBLTGraph, allowing the model to focus on the structural information and node features rather than the arbitrary node ordering. Please refer to Figure 2 for an illustration of the model.

IV. Results and Discussion

The results achieved through the 3-fold cross-validation are presented in Figure 1. As shown by the bar plot displaying the average across the folds, there is very little performance variance in between the folds suggesting a robust model and training schedule. The total running time to achieve this was 91 minutes and 14 seconds. However, the majority of this time was used to compute the metrics, as training SuperBLTGraph on all available data only takes about 10 minutes. The RAM usage of 3-fold CV of the model peaked at 713 MiB, which is small enough to run on even a simple laptop. The training time can be optimised by batching and using GPU capabilities, but considering the low data volume at hand this was not deemed necessary. Batching could also work for regularisation purposes, allowing for more stable gradient descent steps and the inclusion of batch norm layers, so this is certainly an avenue to explore in future work.

To create our final predictions, we retrained the model on 85% of the training set, leaving out 15% for validation. Keeping a validation set was essential for implementing effective learning rate scheduling based on plateauing validation MAE, as well as early stopping. Our Kaggle MAE score on the hidden test set is: 0.148147 which resulted in a 1^{st} position.

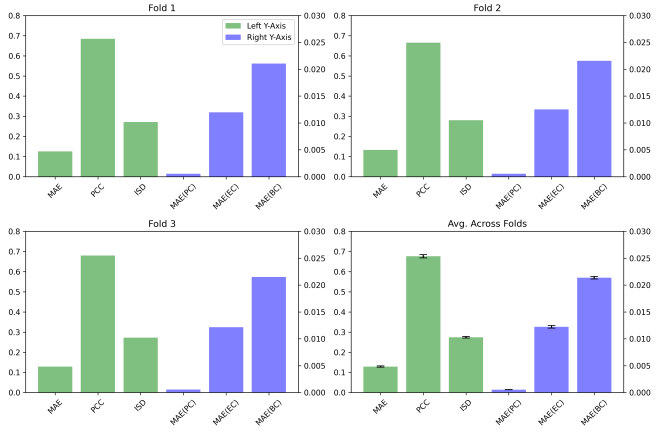

Fig. 1. Results from the 3-Fold Cross-Validation. Please note that the blue and green bars have different y-axes.

REFERENCES

[1] Y. Chen, Y. Xie, Z. Zhou, F. Shi, A. G. Christodoulou, and D. Li, "Brain mri super resolution using 3d deep densely connected neural networks," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 739–742.

[2] F. Shi, J. Cheng, L. Wang, P.-T. Yap, and D. Shen, "Lrtv: Mr image super-resolution with low-rank and total variation regularizations," *IEEE transactions on medical imaging*, vol. 34, no. 12, pp. 2459–2466, 2015.

[3] I. Mhiri, A. B. Khalifa, M. A. Mahjoub, and I. Rekik, "Brain graph super-resolution for boosting neurological disorder diagnosis using unsupervised multi-topology connectional brain template learning," *Medical Image Analysis*, p. 101768, 2020.

[4] M. Isallari and I. Rekik, "Gsr-net: Graph super-resolution network for predicting high-resolution from low-resolution functional brain connectomes," in *Machine Learning in Medical Imaging: 11th International Workshop, MLMI 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4, 2020, Proceedings 11*. Springer, 2020, pp. 139–149.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[6] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Transactions on computational imaging*, vol. 3, no. 1, pp. 47–57, 2016.

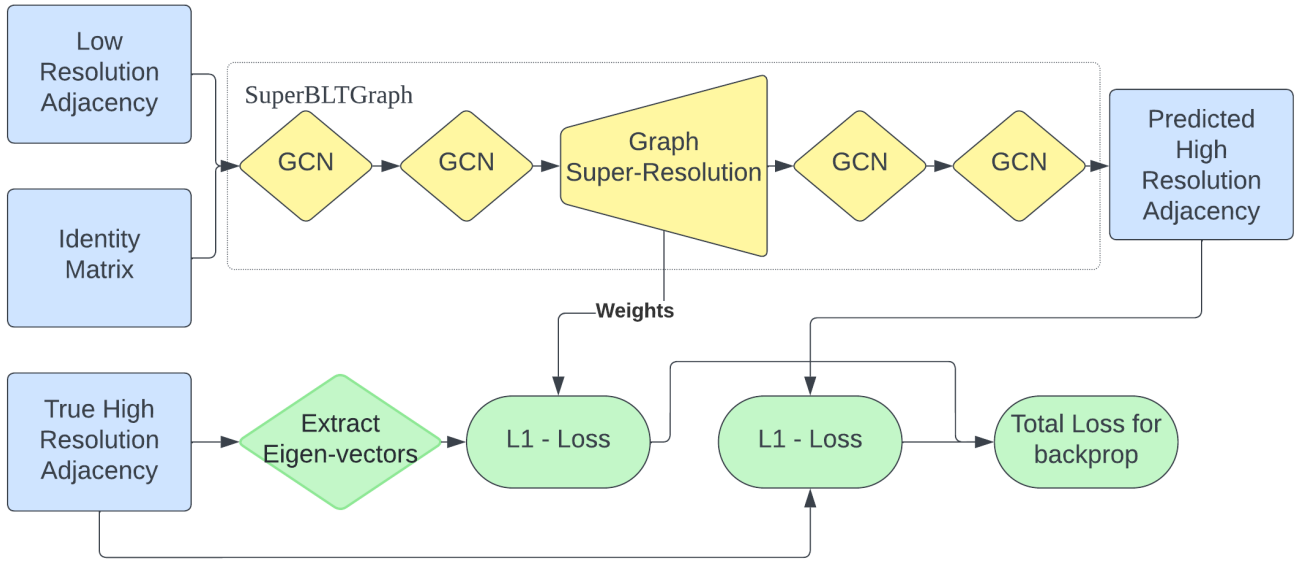

Fig. 2. A figure displaying the architecture of SuperBLTGraph. As shown, the low resolution connectivity graph, together with an identity feature matrix, is passed through a GCN layer, a GSR Layer and subsequently two more GCN layers. The total loss is a combination of the high resolution reconstruction loss, with L1 loss of the high resolution matrix egienvectors and then GSR layer weights.

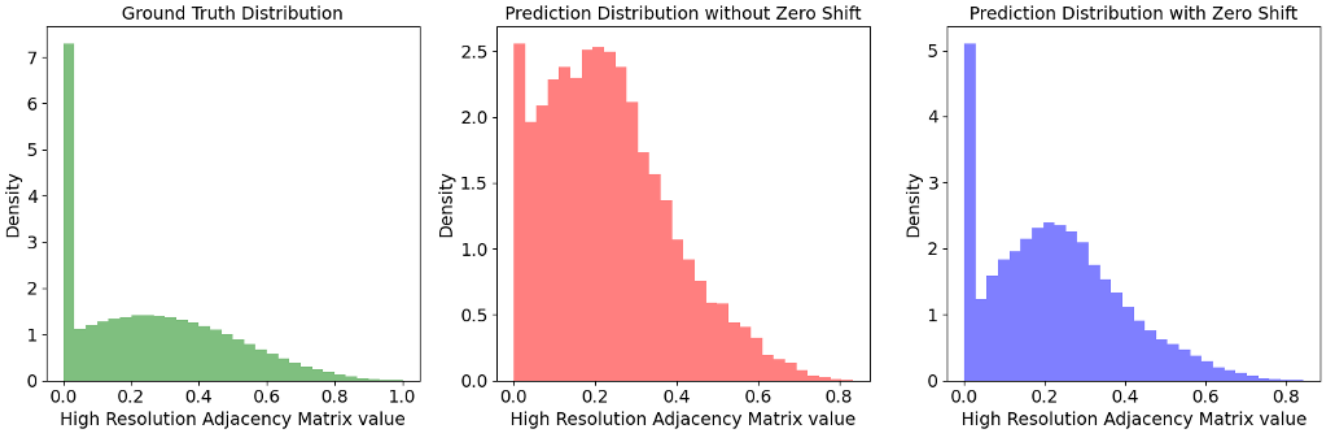

Fig. 3. A figure displaying the ground truth distribution of target adjacency matrix values, as well as the distributions of predicted adjacency matrix values by SuperBLTGraph with and without our proposed zero-shifting. It is evident that this simple pre-processing step, drastically improves the models ability reproduce the target distribution.