CS-E4740 - FEDERATED LEARNING COURSE PROJECT (REVISED): EMPIRICAL GRAPH-BASED FEDERATED LEARNING: A STOCHASTIC BLOCK MODEL APPROACH FOR DISTRIBUTED NETWORK STRUCTURES

ABSTRACT

This paper presents an application of a graph-based federated learning framework to address the challenges of distributed learning and data privacy. We construct an empirical graph using a Stochastic Block Model (SBM) to model the federated learning problem, capturing diverse community structures in the network. Each node maintains a local dataset, synthesized from standard distributions, and a local linear regression model with regularization. These models are trained using two federated learning algorithms - FedSGD and FedAvg. Experimental results demonstrate the efficacy of our framework, with the FedAvg algorithm outperforming FedSGD in terms of average test accuracy. While promising, we also highlight the limitations of the current approach, suggesting potential avenues for future research to improve performance and adapt to dynamic network structures.

***Index Terms*—** Federated Learning, Graph-based Learning, Stochastic Block Model (SBM)

1. INTRODUCTION

Federated Learning is a machine learning paradigm designed to train models directly on devices where data is generated, without needing to centralize the data. This strategy is particularly effective for preserving privacy and reducing the need for data transmission, which can be a substantial benefit for devices with limited bandwidth or in situations where privacy is paramount.

Several researchers have shed light on the application of federated learning within various domains. References [1, 2] provide an exhaustive review of how federated learning can be implemented in the context of Electronic Health Records (EHR) data for healthcare applications and smart healthcare systems. Furthermore, federated learning finds its efficacy in network traffic anomaly detection [3, 4, 5], demonstrating superior performance compared to other data-reconstruction-based detection methodologies. The realm of Edge Computing (EC) also benefits from the application of federated learning, offering solutions to challenges related to unexpected bandwidth loss, data privacy, and legislation [6, 7, 8, 9]. Moreover, federated learning has proven its utility in the domain of recommendation systems by effectively addressing the item cold-start issue [10, 11, 12].

A potential real-life scenario could be in the healthcare industry: a network of hospitals, each possessing a dataset of patient medical records. Privacy regulations often make the centralization of this data unfeasible or illegal. However, the collaborative learning could potentially enhance patient care, as a machine learning model trained on data from all hospitals could be more accurate in predicting patient outcomes compared to a model trained only on data from a single institution.

Here, federated learning can serve as a crucial solution. Through FL, each hospital can independently train a local model on its own data and then communicate model updates with a central server that amalgamates these updates to formulate a global model. Consequently, all participating hospitals can benefit from the collective learning process without needing to share sensitive patient data.

The participating hospitals constitute a part of a larger network, and the relationships among them can be illustrated as a graph. For instance, hospitals may have stronger connections if they are geographically proximate or if they specialize in similar domains. This network of connections can be represented using a Stochastic Block Model (SBM) graph. Furthermore, in this network, it might be advantageous for connected hospitals (i.e., those with frequent patient referrals to each other) to employ similar models. This can be promoted by the regularization term in the loss function you provided, which imposes a penalty on the ℓ_2 norm of the difference between the weights of connected nodes. By employing this graph-based federated learning approach, hospitals can collaboratively train a machine learning model that respects privacy regulations, mirrors the structure of their network, and potentially improves patient care.

The structure of this paper is as follows. Section 2 presents the problem formulation with an SBM graph being the empirical graph along with the synthetic dataset. Section 3 elucidates the methods selected to address this federated learning problem. The performance of these methods on the defined dataset is showcased in Section 4. The paper concludes with the final section.

2. PROBLEM FORMULATION

2.1. Data description

We begin with the conceptualization of an empirical graph, depicted as an undirected and unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Here, nodes $\mathcal{V} := 1, ..., N$ correspond to local datasets $\mathcal{D}^{(i)}$, where $i \in \mathcal{V}$. In the present work, the empirical graph was formed utilizing the Stochastic Block Model (SBM). Each node $i \in \mathcal{V}$ in the empirical graph $\mathcal{G}$ is tied to an independent local dataset $\mathcal{D}^{(i)}$. For clarity, let's denote a local dataset $\mathcal{D}^{(i)}$ as

$$\mathcal{D}^{(i)} = (\mathbf{x}^{(i,1)}, y^{(i,1)}), ..., (\mathbf{x}^{(i,m_i)}, y^{(i,m_i)}), \tag{1}$$

where $\mathbf{x}^{(i,j)}$ and $y^{(i,j)}$ represent the feature and label of the $j-$th data point in the local dataset $\mathcal{D}^{(i)}$, respectively, for $j = 1, \ldots, m_i$. The size m_i of the local dataset varies across distinct nodes $i \in \mathcal{V}$. It is noteworthy to mention that these datasets were synthetically generated from a known distribution. The number of data points at each node ranges from 10 to 100, culminating in a total of approximately 5500 to 10000 data points for our experiments, depending on the specific instance of the SBM.

Furthermore, we would like to clarify that, due to the synthetic nature of our data, feature selection was not required in this case as we directly generated the necessary feature vectors for our study. We acknowledge the importance of feature selection in real-world scenarios and will certainly consider it in future work involving real data.

2.2. Formulation in the form of GTVmin

We employ Alg. 1. FedSGD for Local Linear Regression and Alg. 2. FedAvg outlined in [13] to solve (2) and (3), respectively.

Alg. 1. FedSGD for Local Linear Regression. We solve unique cases of graph total variation minimization (GTVMin)

$$\hat{\mathbf{w}} \in \operatorname*{argmin}_{\mathbf{w} \in \mathcal{W}} \sum_{i \in \mathcal{V}} L_i(\mathbf{w}^{(i)}) + \lambda \sum_{i,i' \in \mathcal{E}} \mathbf{A}_{i,i'} \phi(\mathbf{w}^{(i)}, \mathbf{w}^{(i')}), \tag{2}$$

where the penalty function is defined as $\phi(\mathbf{u}) = \|\mathbf{u}\|_2^2$.

Alg. 2. FedAvg for Linear Regression. We aim to solve GTVMin

$$\begin{aligned} &\hat{\mathbf{w}} \in \operatorname*{argmin}_{\mathbf{w} \in \mathcal{C}} \sum_{i \in \mathcal{V}} L_i(\mathbf{w}^{(i)}), \\ &\mathcal{C} = \mathbf{w} : \mathbf{w}^{(i)} = \mathbf{w}^{(i')} \text{ for any edge } i, i' \in \mathcal{E}. \end{aligned} \tag{3}$$

2.3. Construction of the Empirical Graph

The stochastic block model (SBM) is a generative model for random graphs. It is a flexible model that can generate graphs with communities, or clusters of nodes, that have more connections within each community than between communities.

In our problem, we generate the empirical graph using the SBM, which allows us to model a real-life situation where nodes (in our case, datasets) are organized into a certain number of communities or classes, denoted by $c = 1, \ldots, |\mathcal{C}|$, with $|\mathcal{C}|$ representing the number of classes.

Each community consists of a set of nodes. The total set of nodes across all communities, $\mathcal{V}$, can be written as $\mathcal{V} = i_c$, where i_c ranges from 1 to N across each community c, and c ranges from 1 to $|\mathcal{C}|$.

Specifically, in our application, we define the number of nodes as `n_nodes = 100`, and the number of communities or classes as `n_classes = 10`. We divide the nodes equally among the communities, yielding `sizes = [n_nodes // n_classes] * n_classes`. Here, sizes is a list that describes the size of each community. Each community has an equal number of nodes due to our configuration.

We set the probabilities for connections within and between communities as 0.6 and 0.1, respectively. This means that there's a 60% probability of a connection existing between any two nodes within the same community, and a 10% probability of a connection between nodes from different communities. This is encoded in the `probs` matrix which forms part of the input to the SBM.

Finally, we generate our empirical graph using the SBM by invoking

```
G = nx.stochastic_block_model(sizes, probs).
```

The resultant graph, $\mathcal{G}$, comprises nodes representing local datasets, interconnected as per the structure dictated by the SBM parameters. In this model, edges between nodes are not explicitly chosen or weighted, they are generated stochastically based on the SBM parameters.

2.4. Local Dataset Generation

In this empirical setting, we assign each node $i \in \mathcal{V}$ a unique local dataset $\mathcal{D}^{(i)}$. The data for these nodes is synthetically generated from standard normal distributions and subsequently divided into training, validation, and testing subsets. Each node has a number of samples ranging between 10 and a predefined maximum limit set at `max_samples_per_node = 100`.

Every sample contains a feature vector of length specified by `n_features = 20`, and a label that denotes the community to which the node belongs in the SBM. The labels are integers ranging from 0 to `n_classes - 1` (in this case, 0 to 9), with each label corresponding to a particular community in the SBM.

Local Models. Each node i hosts a same local model, with its parameters denoted by $\mathbf{w}^{(i)}$. The local model is local ridge regression model (linear regression penalized by ℓ_2-

norm). The details will be discussed in Sec. 3. These parameters are learned from the node's local dataset $\mathcal{D}^{(i)}$ and are updated iteratively during the federated learning process.

Features. The features for node i are represented as a matrix $X^{(i)}$, where each row represents a unique sample, and each column represents a feature. These features are synthetically generated from a standard uniform distribution as `X = np.random.rand(m_i,n_features)`, forming the inputs for each local model.

Labels. The labels for node i are assembled in a vector $y^{(i)}$, where each element corresponds to a label for a sample in the local dataset. All nodes that belong to a specific class in the SBM are assigned the same label.

This graph-based federated learning framework is highly adaptable, with the SBM accurately modeling diverse network structures encountered in real-world scenarios. The incorporation of local datasets on each node allows for a distributed, privacy-preserving learning approach.

To summarize the code, we first define the parameters for our SBM graph and our datasets `(n_nodes, n_classes, sizes, intra_prob, inter_prob, n_features, max_samples_per_node)`. We then create a probability matrix probs to represent the probabilities of connections within and between communities.

Using these parameters, we generate the SBM graph and associated datasets using the function `generate_dataset`. Finally, we derive the adjacency matrix `G_Adj` from the SBM graph $\mathcal{G}$. This adjacency matrix will also be used as our empirical graph in the federated learning setting.

3. METHODS

Each node i in our empirical graph hosts a local dataset $\mathcal{D}^{(i)}$ which is represented as $\mathcal{D}^{(i)} = \{\mathbf{X}^{(i)}, \mathbf{y}^{(i)}\}$. Here, features and labels are depicted as

$$\mathbf{X}^{(i)} = \begin{pmatrix} \mathbf{x}_1^{(i)} \\ \vdots \\ \mathbf{x}_{m_i}^{(i)} \end{pmatrix} \in \mathbb{R}^{m_i \times d}, \mathbf{y}^{(i)} = \begin{pmatrix} y_1^{(i)} \\ \vdots \\ y_{m_i}^{(i)} \end{pmatrix} \in \mathbb{R}^{m_i}. \quad (4)$$

We construct a Stochastic Block Model (SBM) with a total of $N = 100$ nodes. Each node i houses a randomly selected number of datapoints $m_i \in [10, 100]$. Our feature selection process employs the standard normal distribution to generate each column in $\mathbf{X}^{(i)}$, rendering them as independent and identically distributed (i.i.d) variables following $N(0, 1)$.

Our choice of local model is an ℓ_2-norm penalized least square regression or linear regression, expressed as:

$$l_i(\mathbf{w}^{(i)})^{A1} = L_i(\mathbf{w}^{(i)}) + \lambda \sum_{i' \in \mathcal{N}^{(i)}} \mathbf{A}_{i,i'} \|\mathbf{w}^{(i)} - \mathbf{w}^{(i')}\|_2^2, \quad (5)$$

$$l_i(\mathbf{w}^{(i)})^{A2} = L_i(\mathbf{w}^{(i)}) + \lambda \|\mathbf{w}^{(i)} - \bar{\mathbf{w}}\|_2^2, \quad (6)$$

$$L_i(\mathbf{w}^{(i)}) = \frac{1}{2} \|\mathbf{y}^{(i)} - \mathbf{X}^{(i)} \mathbf{w}^{(i)}\|_2^2. \quad (7)$$

The ℓ_2-norm regularization term in the above equation quantifies the variation across node weights.

We have utilized the Alg. 1. FedSGD for Local Linear Regression and Alg. 2. FedAvg for Linear Regression described in [13] to train our local models. For Alg. 1., gradient descent is employed to minimize the overall loss function, and the gradient steps for all nodes are performed simultaneously. The ℓ_2-norm of the difference between weights is penalized, pushing the weights towards similarity. The learning rate is calculated before each gradient step, ensuring it is less or equal to the Lipschitz constant of the quadratic term in the local loss function.

The Alg. 2. operates in a server-client paradigm where the server averages all the weights and disseminates this average to clients. Each client then optimizes a loss function, penalized by the difference between its local weight and the average weight. This penalty fosters consistency among local weights.

For model validation, we use a split ratio of 60% training data, 20% validation data, and 20% testing data. The validation process assists in tuning the penalty parameter λ which governs the extent of similarity among node weights.

λ	10^{-3}	10^{-2}	10^{-1}	1
Alg. 1	0.632	0.631	**0.646**	0.619
Alg. 2	**0.843**	0.841	0.840	0.792

Table 1: Validation procedure for hyperparameter tuning. The highest accuracy is marked bold and red.

4. RESULTS

The validation set plays a crucial role in determining the optimal hyperparameter λ, whose potential values lie within the set $[10^{-3}, 10^{-2}, 10^{-1}, 1]$. The corresponding average validation errors for each λ are presented in Table 1. The hyperparameter yielding the highest validation set accuracy is considered the best. As such, $\lambda = 10^{-1}$ is chosen for Alg. 1 and $\lambda = 10^{-3}$ for Alg. 2.

The performances of both algorithms on the dataset are visually represented in Fig .1 and Fig. 2. To enhance clarity in these figures, we randomly sample 50% of all nodes. A small percentage of nodes exhibit low validation and test errors, as depicted in the figures. In contrast, the worst-case scenario results in an accuracy of 0%, indicating no correct feature

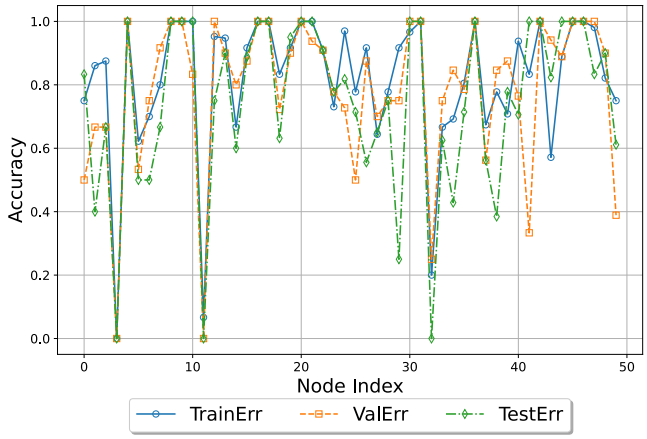

Fig. 1: The performance of Alg. 1 FedSGD on the dataset.

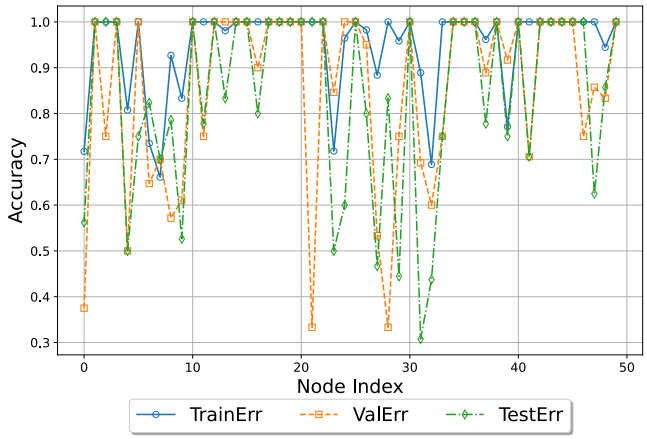

Fig. 2: The performance of Alg. 2 FedAvg on the dataset.

classification. A comparison of Fig. 1 and Fig. 2 reveals that Alg. 2 FedAvg boasts a greater number of nodes with perfect test accuracies, and fewer nodes with poor test accuracies.

To select the superior method, we rely on the average test accuracy as the determining metric. The results are as follows:

1. Alg. 1. FedSGD: 60.2%
2. Alg. 2. FedAvg: 63.1%

Therefore, Alg.2. FedAvg emerges as the method of choice based on these results. Its associated test accuracies are depicted in Fig. 2. The empirical graph, local model, and measure for Alg. 2. FedAvg have been detailed in Sec.2 and Sec. 3.

5. CONCLUSION AND DISCUSSION

In summary, this paper introduces a graph-based federated learning framework applicable to real-world scenarios where network nodes collectively learn a global model while safeguarding data privacy. The empirical graph, constructed using the SBM, effectively simulates the federated learning problem, enabling the generation of a graph with varied community structures. For each node, local datasets comprising synthetic distributions are devised, featuring unique features and labels. Local models, fashioned as linear regression models with regularization, are trained utilizing federated learning algorithms.

The experimental results substantiate the proficiency of the proposed framework. Both Alg. 1 FedSGD and Alg. 2 FedAvg deliver substantial accuracies in feature classification, with Alg. 2 FedAvg surpassing Alg. 1 FedSGD in terms of mean test accuracy. These outcomes imply a substantial resolution of the problem at hand, thus underscoring the potential of graph-based federated learning in distributed and privacy-preserving learning contexts.

Although our methods performed well in most scenarios, we observed that the training error at some nodes was much smaller than the corresponding validation error. This suggests a potential issue of overfitting, where our models may be fitting too closely to the training data and not generalizing well to unseen data. One limitation of our current method is the use of linear regression models, which may not capture complex non-linear relationships in the data. In future work, we plan to explore more advanced machine learning models, such as deep learning, to improve our method's performance. In addition to exploring different machine learning models, we also plan to collect more training data and experiment with different choices for the regularization parameter. We believe these steps will further improve our method's performance and robustness.

6. REFERENCES

[1] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, May. 2022.

[2] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Comput. Surv.*, vol. 55, no. 3, feb 2022.

[3] J. Pei, K. Zhong, M. A. Jan, and J. Li, "Personalized federated learning framework for network traffic anomaly detection," *Computer Networks*, vol. 209, p. 108906, 2022.

[4] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for iot security at-

tacks," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2545–2554, 2022.

[5] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu, "Multi-task network anomaly detection using federated learning," in *Proc. 10th International Symposium on Information and Communication Technology*, ser. SoICT '19, 2019, p. 273–279.

[6] H. G. Abreha, M. Hayajneh, and M. A. Serhani, "Federated learning in edge computing: A systematic survey," *Sensors*, vol. 22, no. 2, 2022.

[7] Y. Ye, S. Li, F. Liu, Y. Tang, and W. Hu, "Edgefed: Optimized federated learning based on edge computing," *IEEE Access*, vol. 8, pp. 209 191–209 198, 2020.

[8] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 806–12 825, 2021.

[9] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[10] O. A. Wahab, G. Rjoub, J. Bentahar, and R. Cohen, "Federated against the cold: A trust-based federated learning approach to counter the cold start problem in recommendation systems," *Information Sciences*, vol. 601, pp. 189–206, 2022.

[11] J. Vyas, D. Das, and S. K. Das, "Vehicular edge computing based driver recommendation system using federated learning," in *Proc. 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2020, pp. 675–683.

[12] Z. Teimoori, A. Yassine, and M. S. Hossain, "A secure cloudlet-based charging station recommendation for electric vehicles empowered by federated learning," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6464–6473, 2022.

[13] A. Jung, "Federated learning," *Lectures of the course CS-E4740, Aalto University*, 2023.

RESPONSE TO THE COMMENTS OF THE REVIEWERS ON MANUSCRIPT:

CS-E4740 - FEDERATED LEARNING COURSE PROJECT:
EMPIRICAL GRAPH-BASED FEDERATED LEARNING: A STOCHASTIC BLOCK MODEL APPROACH FOR DISTRIBUTED NETWORK STRUCTURES

1. COMMENTS OF REVIEWER #1

Q3.2 No, there is no discussion since data is constructed synthetically.
Feedback: Although our dataset is synthetic, the features were selected to create a realistic simulation for our model. We'll make this clear in our revised report.
Q3.4 The selection of the local models is not explained. This is might be related to lack of the specific domain and real data
Feedback: The choice of local models was motivated by their appropriateness for the type of synthetic data used. We'll include a brief explanation in our revised manuscript.
Q3.5 The local loss functions are not discussed.
Feedback: Thanks for this comment. Our model uses a standard loss function for ridge regression, which we thought was self-evident. However, we agree that a brief explanation would enhance the report, and we will include it.
Q3.7 No
Feedback: Our data splitting strategy is a standard 60-20-20 split for training, validation, and testing. We acknowledge that an explicit mention in the report will improve clarity, and we will add this.
Q4.2 No discussion.
Feedback: We appreciate your input on this. The construction of the edges in the empirical graph is based on the Stochastic Block Model parameters. We'll make sure to clarify this in our revision.

2. COMMENTS OF REVIEWER #2

Q3.1 Synthetic dataset, meaning it was synthetically generated from a distribution. Each node has 10 - 100 data points and there were 100 nodes. It is still a bit unclear how many data points were there when you ran it. I think this could have been mentioned. Also it would have been good to have a plot of the data points to see how they were distributed.
Feedback: Thank you for the feedback. I believe the dataset is very clear with the mathematical expression, and as it is a synthetic dataset for quick scaling.
Q3.6 Examples for variation measures include the norm of the difference between local model parameters or the squared difference of their predictions on a common test set.
Feedback: We appreciate your suggestion. The variation of local models was measured using the norm of the difference between local model parameters, as mentioned in the local loss functions section. We will clarify this in our revised manuscript.

3. COMMENTS OF REVIEWER #3

Q3.4 No
Feedback: The choice of local models was driven by the need for simplicity and interpretability in our synthetic dataset scenario. The linear regression model was chosen due to its wide applicability and ease of understanding. It also has the advantage of being computationally efficient, a factor that is crucial in the federated learning setup. We will further motivate this choice in the revised manuscript.
Q3.2 There are description of the feature vector structure but no mention of how the features are selected.
Feedback: Thank you for your comment. The features in our synthetic dataset are generated from a uniform distribution, as mentioned in the manuscript. The selection process wasn't based on domain knowledge, data visualization, or any other strategies, since the data was synthetically created. However, we will provide more clarity on this in our revision.

4. COMMENTS OF REVIEWER #4

Q4.3 Does the report (1) explain how the test set is constructed for each node and (2) clearly state the test error at each node ? A test set should consist of data points that have neither been used to train the local models (training set), nor for choosing between different local models (validation set). The test error is the average loss incurred on a test set. – Yes, the test set selection is well explained. However test errors at each node were not clearly presented in e.g. Table.
Feedback: Thank you for bringing this to our attention. In our current study, we primarily focused on the aggregate performance of the Federated Learning system. We acknowledge that providing a detailed node-wise error analysis could add value.

5. COMMENTS OF REVIEWER #5

Q4.2 Mentioned how they were penalized but noting about the final chosen edges and their weights.

Feedback: We appreciate your feedback. In our report, we focused on discussing the penalization process rather than the final selected edges and their weights. However, we understand that such information could be of interest. We will consider adding more details on the chosen edges and their weights in our future work.

Q4.3 All mentioned but test error was mentioned for the model not for all the nodes.

Feedback: Thank you for your comments. We acknowledge that our report could be improved by providing more detailed information about the test error for each node. In our next revision, we plan to include this analysis for a more comprehensive understanding of our model's performance.