LRGB Project Report

Sara Zatezalo, Justin Manson

***Abstract*— In the field of Graph Neural Networks (GNNs), the message-passing (MP) framework is widely adopted for its ability to propagate information across graph nodes. However, capturing long-range interactions (LRI) in graphs poses challenges such as over-squashing and over-smoothing. This project evaluates several GNN models on the Peptides-functional dataset, part of the Long Range Graph Benchmark (LRGB). We compare the performance of several models, such as a Graph Convolutional Network (GCN), GCN augmented with virtual nodes, GCN augmented with Dynamic Rewiring (DRew), and a Spectral Attention Network (SAN), to determine their effectiveness in handling long-range dependencies. As an additional sanity check, we explore an alternative classification framework that operates independently of the graph topology, namely attention-based Multi Instance Learning (MIL). Our findings highlight the limitations of traditional MP frameworks for modelling LRIs, and show how innovations such as dynamic rewiring and attention mechanisms that allow more direct propagation of information across graphs can yield significant improvements. We have open-sourced all code related to our work on GitHub.**

I. INTRODUCTION

In the field of Graph Neural Networks (GNNs), the Message-Passing (MP) framework [1] is a notably represented and widely adopted approach due to its ability to collect and propagate information across nodes and edges of a graph, enabling learning of graph-structured data. During each message-passing iteration in a GNN (i.e., each layer), a hidden embedding corresponding to every node is updated, using the aggregated information from its 1-hop neighborhood. As these iterations progress, each node representation contains more and more information from further reaches of the graph. Precisely, after *k* layers of a Message Passing Neural Network (MPNN), every node contains information about its *k*-hop neighborhood [2].

In some learning tasks, capturing long-range interactions (LRI) in graphs, i.e. information from distant nodes, is crucial. Several important features characterize LRI graphs. The size of the graph, the contribution of its global structure, and the nature of the task all play significant roles in determining long-range dependencies [3]. If for a certain LRI task an *L*-hop neighborhood information is needed, the minimal number of layers required for an MPNN to capture it is *L*. As the amount of information to be aggregated grows exponentially, such as the neighborhood of each node, this raises several issues, commonly referred to as *over-squashing* and *over-smoothing*. The former occurs when information from distant nodes is integrated through edges that are graph bottlenecks. At the same time, the latter represents the issue of embeddings of local nodes being similar, although they might belong to different classes [4].

Recently, multiple methods have been developed to mitigate the above-mentioned issues and improve the learning performance of LRI graphs. Many of them are known as 'graph rewiring' methods. 'Local' graph rewiring methods rely on adding elements to increase connectivity, such as global nodes [1] or global layers, i.e. structural and positional encoding [5], or aggregating over multiple hops at each layer [6]. On the other hand, by their design, Graph Transformers can leverage full connections of the input graphs and capture long-range dependencies, thus preventing the information bottleneck [3]. Drawing inspiration from [7], we refer to these as 'global' graph rewiring methods.

In this project, we test several GNN models and compare their performance on a classification task using the `Peptides-func` dataset, to check their capability to capture long-range interactions. To fairly evaluate the models and observe the difference in effectiveness, it is paramount to choose the correct dataset. It is often the case that the models are tested on datasets where tasks rely on local information, thus attempts to solve issues regular MPNNs exhibit are shown to be insignificant. `Peptides-func` dataset is part of a carefully constructed Long Range Graph Benchmark (LRGB) [3], which contains real-world data with LRI. In their work, Dwivedi et al. [3] show that fully connected models outperform local MPNNs on proposed datasets, making them eligible to explore and test other LRI architectures on them.

II. NOTATION

We denote a graph by $G = (V, E)$ where V is the set of N nodes and E is the set of edges. The adjacency matrix of the graph is represented by $A \in \mathbb{R}^{N \times N}$ and the degree matrix D is a diagonal matrix with $D_{ii} = \sum_j A_{ij}$. Let $X \in \mathbb{R}^{N \times F}$ be the matrix of node features, where F is the number of input features per node. The graph Laplacian is defined as $L = D - A$. A normalized version of the graph Laplacian is given by $L = I_N - D^{-1/2} A D^{-1/2}$, where I_N is the identity matrix.

The eigenvalues and eigenvectors of the Laplacian are denoted by λ_i and ϕ_i, respectively. The learned positional encoding (LPE) of node i is denoted by PE_i.

III. BACKGROUND

The following section provides a detailed outline of a select group of classification model architectures. We start by detailing Graph Convolutional Networks (GCNs), a widely used message-passing graph neural network architecture, which serves as a baseline in our experiments. We then outline two approaches: Spectral Attention Networks (SAN)

and Dynamically Rewired Message Passing (DRew), that aim at addressing common limitations of MPNNs for modeling LRIs. Finally, we present an Attention-based Multi Instance Learning (MIL) model, which we employ as a sanity check to justify the use of network-based classification models for modeling LRIs.

A. Graph Convolutional Networks

Graph Convolutional Networks (GCNs) [8] are a type of neural network that operates directly on graphs. They extend the concept of convolutional neural networks to graph-structured data by defining convolutions in the spectral domain. The key idea is to apply a filter to the spectral representation of the graph.

The layer-wise propagation rule for a GCN is defined as:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right),$$

where $\tilde{A} = A + I_N$ is the adjacency matrix with added self-connections, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$ is the degree matrix of $\tilde{A}$, $H^{(l)}$ is the matrix of activations in the l-th layer with $H^{(0)} = X$, $W^{(l)}$ is the layer-specific trainable weight matrix, and $\sigma(\cdot)$ is an activation function, such as ReLU.

This framework allows the GCN to learn node representations by leveraging both the graph structure and the node features, making it effective for tasks such as node classification and link prediction.

B. Spectral Attention Networks

Spectral Attention Networks (SANs) [9] are designed to generalize the Transformer architecture to graph-structured data by leveraging the spectral properties of the graph Laplacian. The SAN model incorporates Learned Positional Encodings (LPEs) that utilize the full Laplacian spectrum to capture the positional information of nodes within the graph.

a) Learned Positional Encodings (LPEs): SANs use the eigenvectors ϕ_i of the graph Laplacian as positional encodings. Each node i is assigned a positional encoding PE_i derived from the eigenvectors and eigenvalues of the Laplacian. Specifically, the positional encoding for node i is given by concatenating the first m eigenvectors weighted by their corresponding eigenvalues:

$$PE_i = [\lambda_1\phi_{1,i}, \lambda_2\phi_{2,i}, \ldots, \lambda_m\phi_{m,i}],$$

where $\phi_{k,i}$ denotes the i-th element of the k-th eigenvector.

b) Transformer Encoder: The node features X and the positional encodings PE are combined and passed through a Transformer encoder. The Transformer encoder employs multi-head self-attention mechanisms to allow nodes to attend to all other nodes in the graph, effectively capturing long-range dependencies. The input to the Transformer encoder is:

$$H^{(0)} = X + PE,$$

where $H^{(0)}$ is the initial node representation matrix.

c) Attention Mechanism: The attention weights are computed using the scaled dot-product attention mechanism. For nodes i and j, the attention coefficient e_{ij} is calculated as:

$$e_{ij} = \frac{(Qh_i) \cdot (Kh_j)^T}{\sqrt{d_k}},$$

where Q and K are the query and key projection matrices, respectively, h_i and h_j are the feature vectors of nodes i and j, and d_k is the dimension of the key vectors.

d) Fully Connected Graph: To mitigate the over-squashing problem common in GNNs, SANs fully connect the graph by adding edges between all pairs of nodes. This ensures that information can flow freely between any two nodes in the graph.

The SAN model is summarized in Figure 1 from the original paper, which illustrates the process of generating LPEs, combining them with node features, and processing them through the Transformer encoder.

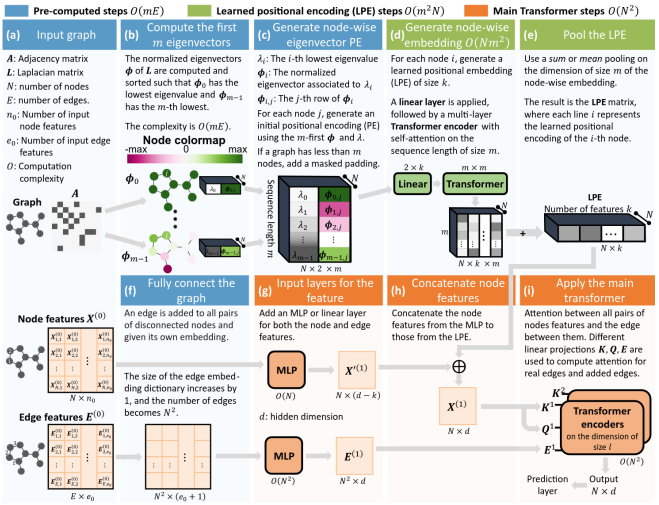

Fig. 1: The proposed SAN model with the node LPE, a generalization of Transformers to graphs.

C. Dynamically Rewired Message Passing

Dynamically Rewired Message Passing (DRew) [7] is a novel framework designed to improve the message-passing process in GNNs by addressing the limitations of classical MPNNs such as over-squashing and over-smoothing.

Classical MPNNs rely on local aggregation where information is passed between immediate neighbors. This setup can lead to the phenomena of over-squashing, where information from distant nodes is compressed through a limited number of connections, and over-smoothing, where features become indistinguishable as the number of layers increases.

To address these issues, DRew introduces a layer-dependent rewiring mechanism that gradually increases the connectivity of the graph, allowing for more efficient information flow across distant nodes without losing the inductive bias provided by the original graph structure. The rewiring process is defined as follows:

$$a_{i,k}^{(\ell)} = \text{AGG}_k^{(\ell)}\left(\{h_j^{(\ell)} : j \in \mathcal{N}_k(i)\}\right), \quad 1 \le k \le \ell + 1$$

$$h_i^{(\ell+1)} = \text{UP}_k^{(\ell)}\left(h_i^{(\ell)}, a_{i,1}^{(\ell)}, \ldots, a_{i,\ell+1}^{(\ell)}\right)$$

where $\mathcal{N}_k(i)$ denotes the k-hop neighborhood of node i, AGG represents the aggregation function, and UP represents the update function. At each layer ℓ, the receptive field of node i expands by one hop, allowing distant nodes to interact directly as the network grows deeper.

Additionally, DRew introduces a delay mechanism controlled by a parameter ν, which allows for skip connections based on the mutual distance between nodes. This mechanism is defined as νDRew and helps mitigate over-smoothing by delaying the aggregation of messages from distant nodes:

$$a_{i,k}^{(\ell)} = \text{AGG}_k^{(\ell)}\left(\{h_j^{(\ell-d)} : j \in \mathcal{N}_k(i)\}\right), \quad 1 \le k \le \ell + 1$$

where d is the delay introduced by the parameter ν.

D. Attention-based Multi Instance Learning

Attention-based Multi Instance Learning (MIL) [10] is a framework where a bag of instances is associated with a single label, and the model's goal is to predict this label while identifying key instances contributing to the decision. Gated attention-based aggregation is a flexible and interpretable method to combine these instances into a single representation.

Given a bag of K instances represented by their embeddings $H = \{h_1, h_2, \ldots, h_K\}$, the aggregated representation z is computed as a weighted sum of these embeddings, where the weights are determined using an attention mechanism. The attention score a_k for instance k is computed as:

$$a_k = \frac{\exp\left(w^\top\left(\tanh\left(Vh_k^\top\right) \odot \sigma\left(Uh_k^\top\right)\right)\right)}{\sum_{j=1}^{K}\exp\left(w^\top\left(\tanh\left(Vh_j^\top\right) \odot \sigma\left(Uh_j^\top\right)\right)\right)}$$

Here, $V \in \mathbb{R}^{L\times M}$ and $U \in \mathbb{R}^{L\times M}$ are learnable parameters, $w \in \mathbb{R}^{L\times 1}$ is a weight vector, $\tanh(\cdot)$ is the hyperbolic tangent function, $\sigma(\cdot)$ is the sigmoid function, and $\odot$ denotes element-wise multiplication. This gating mechanism introduces an additional non-linearity, making the model more expressive.

The final aggregated representation z is then given by:

$$z = \sum_{k=1}^{K} a_k h_k$$

In the context of graphs, this method can be applied by treating the feature vector of each node as an instance in a bag. The gated attention mechanism aggregates these node features into a single embedding, capturing the most informative aspects of the graph structure and node features.

Since we are applying this framework to a graph classification task which requires long range interaction modelling, a potential advantage of this method is that it completely ignores graph topology, and hence circumvents the need to model long range interactions, directly grouping important but distant elements of the graph into a single embedding. Additionally, this approach is significantly simpler, requiring fewer parameters than a typical deep GNN.

IV. Limitations of LRI Modelling

A. Over-squashing

MPNNs have great success in learning graph-structured data, gathering information through interconnected nodes, thus capturing complex relationships. However, when it comes to long-range interactions, they exhibit a challenge called over-squashing. This issue occurs because of the rapid (exponential) growth of a node's receptive field, with the increasing length of the network, which results in the compression of a large number of messages into fixed-size vectors (embeddings) [4]. This causes the incapacity of input features of distant nodes to influence a node representation at the final layer, ultimately leading to the loss of information.

B. Over-smoothing

Over-smoothing in GNNs occurs when embeddings of nodes from different classes become indistinguishable. This problem arises in multi-layer MPNNs, when the task relies on short-range interactions between nodes in the local neighborhood. However, with increasing layers of the network, trying to capture information from distant nodes, the repeated aggregation of local information causes the embeddings to lose their capacity to distinguish between nodes belonging to different classes [4].

C. Handling Limitations in GNNs

To overcome the above-mentioned issues, the literature has proposed several strategies. Many of them fall under the category of *graph-rewiring*, which modifies the structure of the graph by adding/removing edges to optimize the flow of information. Examples include introducing virtual nodes [11], Cayley graphs [12], adding a global layer such as position/structural encoding [5], or a very straightforward idea of including a fully-adjacent matrix at the final GNN layer, connecting every pair of nodes [13].

However, these rewiring methods come with certain drawbacks, such as the loss of valuable topological information, i.e. losing the inductive bias provided by distance on graph [7], and the smoothing effect on the graph. To overcome the issue of over-smoothing and over-squashing together, graph transformers are now widely being utilized for their ability to establish direct connections between distant nodes. Still, this introduces their main limitation which is losing the notion of locality, which has to be introduced with positional or structural embeddings [7]. Additionally, their computational and memory demands are substantial because each node is connected to all other nodes.

V. EXPERIMENTS

A. Dataset

The `Peptides-functional` dataset is part of the Long Range Graph Benchmark (LRGB) and consists of 15,535 graphs, each representing a peptide, which is a short chain of amino acids [3]. The nodes in these graphs correspond to the heavy (non-hydrogen) atoms of the amino acids, and the edges represent the bonds between these atoms. This structure means each graph encodes the sequence of amino acids without incorporating higher-dimensional structural information, emphasizing the importance of recognizing local structures and the positions of amino acids within the sequence.

On average, the graphs in this dataset contain 150.94 nodes and have a diameter of 56.99. This large graph size and significant diameter make this dataset well-suited for testing the capabilities of various GNN architectures in capturing LRIs. Furthermore, this ensures that any graph positional or structural encoding used must generalize well across various graph sizes and be computationally efficient.

The task associated with the `Peptides-functional` dataset is multi-label classification. There are 10 functional classes, including categories such as Antibacterial, Antiviral, and cell-cell communication, among others. Each peptide can belong to multiple classes simultaneously, with an average of 1.65 classes per peptide. The class distribution is highly imbalanced, with only 16.5% of the data points being positive examples in any given class. The richest class has 62.7% positives, while the poorest has only 1.9%.

To train the classification model, we use the *PyTorch Geometric* integration of the LRGB [14]. This library facilitates the loading and processing of the `Peptides-functional` dataset. The dataset is split into training, validation, and test sets using stratified splitting based on meta-classes, ensuring balanced representation across the splits. The ratio for the splits is 70% for training, 15% for validation, and 15% for testing.

In training the classification model, we use a multi-label binary cross-entropy loss function, which is suitable for tasks where each instance can belong to multiple classes. The model's performance is evaluated using the unweighted mean Average Precision (AP). This metric measures the area under the precision-recall curve and is particularly effective for datasets with imbalanced classes.

B. Results and Discussion

In this section, we discuss the performance of various GNN architectures tested on the `Peptides-functional` dataset. Our results are summarized in Table I.

Each model's performance can be primarily attributed to its approach in addressing the long-range interaction (LRI) challenges posed by the graph structure of the dataset. Standard Graph Convolutional Networks (GCN) achieved an AP of 0.5799, suggesting moderate effectiveness in capturing the complex patterns of LRIs. The introduction of virtual nodes in the GCN model, where we added 10 virtual nodes with random edge weights and preserved the original edge weights as unity, significantly enhanced the model's capacity to encapsulate distant node information, improving AP to 0.6702.

TABLE I: Comparison of Average Precision for Different Architectures

Architecture	**Average Precision (AP)** ↑
Attention Pooling	0.5256
Multi-Headed Attention Pooling	0.5837
Graph Convolutional Network (GCN)	0.5799
GCN with Virtual Nodes	0.6702
GCN with Dynamic Rewiring (DRew)	0.6692
Spectral Attention Network (SAN)	0.6620

The Dynamically Rewired Message Passing (DRew) model, which directly augments the graph connectivity per layer, exhibited a comparable performance with an AP of 0.6692. This model facilitates a more efficient transfer of information across multi-hop nodes, a clear advantage in dense and large graphs prevalent in our dataset.

Spectral Attention Networks (SAN), using a configuration as specified in the original papers [9], performed robustly with an AP of 0.6620. SAN models leverage attention mechanisms to form a fully connected graph, disregarding the initial graph structure.

Interestingly, the simpler Multi-Instance Learning (MIL) classifiers, which aggregate node features using an attention mechanism without considering the graph topology, achieved substantial results. The base MIL model, employing attention pooling, scored an AP of 0.5256, while its enhanced version with 10 attention heads increased this score to 0.5837, thus out-performing the significantly more complex GCN model.

From these results, we observe that methods mitigating the over-squashing and over-smoothing phenomena, such as DRew and the use of virtual nodes, tend to outperform standard GNN approaches. These innovations enable better modeling of LRIs by either providing new interaction pathways (virtual nodes) or by enhancing direct message passing across nodes (DRew).

VI. CONCLUSION

In conclusion, our work demonstrates the potential of various graph neural network architectures to handle long-range interactions. Innovations such as virtual nodes and DRew significantly enhance model performance by addressing the inherent challenges of over-squashing and over-smoothing in standard message-passing frameworks. The SAN, by leveraging full connectivity, also shows promising results, highlighting the effectiveness of attention mechanisms in GNNs. Surprisingly, the simple attention-based MIL model demonstrates comparable results to the more complex GCN in this task, suggesting that the challenge of modeling LRIs with GNNs might warrant exploring alternative classification approaches that bypass the need to propagate information across lengthy edge chains.

REFERENCES

[1] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," 2017.

[2] W. L. Hamilton, "Graph representation learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159.

[3] V. P. Dwivedi, L. Rampášek, M. Galkin, A. Parviz, G. Wolf, A. T. Luu, and D. Beaini, "Long range graph benchmark," 2023.

[4] S. Akansha, "Over-squashing in graph neural networks: A comprehensive survey," 2024.

[5] V. P. Dwivedi, A. T. Luu, T. Laurent, Y. Bengio, and X. Bresson, "Graph neural networks with learnable structural and positional representations," 2022.

[6] S. Abu-El-Haija, A. Kapoor, B. Perozzi, and J. Lee, "N-gcn: Multi-scale graph convolution for semi-supervised node classification," 2018.

[7] B. Gutteridge, X. Dong, M. Bronstein, and F. D. Giovanni, "Drew: Dynamically rewired message passing with delay," 2023.

[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017.

[9] D. Kreuzer, D. Beaini, W. L. Hamilton, V. Létourneau, and P. Tossou, "Rethinking graph transformers with spectral attention," *CoRR*, vol. abs/2106.03893, 2021.

[10] M. Ilse, J. M. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," *CoRR*, vol. abs/1802.04712, 2018.

[11] C. Cai, T. S. Hy, R. Yu, and Y. Wang, "On the connection between mpnn and graph transformer," 2023.

[12] A. Deac, M. Lackenby, and P. Veličković, "Expander graph propagation," 2022.

[13] U. Alon and E. Yahav, "On the bottleneck of graph neural networks and its practical implications," 2021.

[14] P. G. Team, *LRGBDataset*, 2023. Accessed: 2024-06-10.