Graph learning for capturing long-range dependencies in protein structures

Ali Hariri
Signal Processing Laboratory (LTS2)
École Polytechnique Fédérale de Lausanne (EPFL)
ali.hariri@epfl.ch

Pierre Vandergheynst
Signal Processing Laboratory (LTS2)
École Polytechnique Fédérale de Lausanne (EPFL)
pierre.vandergheynst@epfl.ch

Abstract

Polyamides, or peptides and proteins, are biomolecules that exist in a broad spectrum of size, structure, and function. Both structure and function are defined by the underlying sequence of amino acids, causing the polyamide to take three-dimensional conformations when in solution. Despite significant efforts and advances in function and conformation prediction, there remains a critical need for computational methods to accurately infer protein function from sequence and structure. Recent advancements in deep learning, particularly Graph Neural Networks, have shown promise in learning the sequence and structure of proteins. However, they fail to capture essential long-range dependencies inherent in the complex and dynamic three-dimensional structures of proteins, leading to issues including oversquashing and oversmoothing. Here, we explore solutions to the challenge of capturing long-range dependencies in graph representations of polyamides, focusing on latent nodes and graph rewiring techniques. While graph rewiring enhances information flow between distant nodes, latent nodes enable the concentration of global information. In addition, we investigate the effectiveness of ChebNet, a spectral backbone, in capturing long-range dependencies. Our unified framework combines these approaches to address the limitations of current methods, offering insights into protein function and regulation. Through experimental analysis, we demonstrate the efficacy of our proposed methods in capturing long-range dependencies.

1 Introduction

Proteins are biomacromolecules that serve as essential components within cells and play critical roles in nearly every biological process, including catalyzing metabolic reactions, replicating DNA, and transporting molecules. They are comprised of a sequence of amino acids each of which possesses a distinct side chain, leading to an incredibly vast array of potential protein sequences. Protein functions determine health outcomes and the progression of diseases, hence predicting the functional properties of proteins is vital for developing new drug therapies. Despite the considerable expense and time required for function annotation of new protein sequences, there's a pressing need for accurate and efficient in-silico methods to bridge the gap between sequence and function. However, sequence-based approaches do not directly incorporate or utilize known structural information, which is crucial for understanding protein functions. Protein design has emerged as an integral aspect of pharmaceutical research. Current efforts seek to better understand the design principles that form a basis for the structure and functions of proteins. This would enable the discovery of proteins with properties that are key for therapeutic and technological applications.

Recent advances in deep learning paved the way for new methods in protein design. In particular, Graph Neural Networks (GNNs) [1, 2] have emerged as a powerful tool for learning structural representations of proteins and biomolecules [3, 4]. GNNs are a class of machine learning models designed to operate on graph-structured data and extract information by iteratively aggregating and updating node features based on their local neighborhood connections. Despite their general success, GNNs exhibit clear limitations when confronted with long-range dependencies in graph learning tasks. In such scenarios, GNNs are prone to the phenomenon described by Alon and Yahav as **oversquashing**, whereby the propagated information along K graph layers are 'squashed' into fixed sized vectors as the receptive field of a given node grows rapidly [5]. Another problem arises when the number of layers of a GNN increases, resulting in a loss of discriminative information and convergence of node embeddings towards similar values. This phenomenon is

referred to as **oversmoothing** [6, 7, 8]. These two phenomena diminish the expressive power of current graph-based architectures, especially on larger graphs exhibiting long-range dependencies. The latter are fundamental aspects of proteins' structural and functional complexity, as inferred from their residue interactions. These interactions play pivotal roles in stabilizing tertiary structures, facilitating ligand binding, and orchestrating allosteric regulation [9, 10, 11]. Understanding long-range dependencies is crucial for deciphering protein folding mechanisms, predicting protein structures from sequences, and designing novel therapeutics targeting protein-protein interactions. Therefore, it is crucial to account for such dependencies when modeling proteins with GNNs, rising the need for more expressive architectures that would account for distant re interactions.

To tackle the aforementioned limitations of GNNs, Transformers have emerged as powerful learners on graphs as they alleviate the problems of oversquashing by allowing a given node to attend to all other nodes through global attention modules [12]. However, a few disadvantages are associated with Transformers, mainly their computational cost and the loss of locality and connectivity of a given graph, which has been shown to be essential in graph learning [13]. A set of alternative approaches has been proposed, mainly through the use of virtual nodes to reduce the commute time between any two given nodes [14], or by changing the graph topology to allow for a better flow of information by optimizing some properties related to graph bottlenecks: this is known as **graph rewiring** which has been recently investigated [15, 16, 17]. Despite the aforementioned recent advances in tackling long-range dependencies in graphs, those are still under-explored for learning on protein structures.

1.1 Main contributions

In this paper we shed light on the power of long-range techniques in representing protein structures for diverse tasks at the global (protein) and local (residue) levels.

- We propose a GNN-based rewiring scheme that defines a set of trainable latent nodes to cover different regions of the protein. Our model then uses the latent nodes as mediators to rewire the graph by attending distant nodes through attention-based edge addition.
- We make a pragmatic choice of an expressive and theoretically-grounded spectral backbone, ChebNet, for learning complex protein substructures and long-range dependencies. We empirically show the superiority of this backbone on various protein learning tasks and under different constraints. An analysis for the obtained improvement is given in Section 6.1

2 Background

2.1 Definitions

Let $G = (V, E)$ be a graph with a set of n nodes V and edges E encoded in the adjacency matrix $A \in \mathbb{R}^{n\times n}$ describing the graph's connectivity. Another essential matrix associated to the graph is its Laplacian: Given the diagonal degree matrix $D_{ii} = \sum_j W_{ij}$ of the graph, we can obtain its combinatorial Laplacian $L = D - A$ and the normalized form $L_{norm} = I_n - D^{\frac{-1}{2}} A D^{\frac{-1}{2}}$, which is real, symmetric and positive semi-definite. The eigendecomposition of the Laplacian reads $L = U\Lambda U^T$ and results in a set of orthonormal eigenvectors $U = [u_0, ..., u_{n-1}] \in \mathbb{R}^{n\times n}$ and the associated eigenvalues $\Lambda = diag([\lambda_0, ..., \lambda_{n-1}]) \in \mathbb{R}^{n\times n}$ are known as the Graph Fourier modes and their frequencies, respectively.

2.2 Message Passing Neural Networks

Let $X \in \mathbb{R}^{n\times d}$ be the matrix encoding the node features of the graph G. Message Passing Neural Networks (MPNNs) typically use functions that take X and A as input and use them to aggregate features from the neighborhood of a given node to update its representation in a high-dimensional space. A typical example of an MPNN can be found in [18]. The node feature update after one layer is given by:

$$h_i^{(l+1)} = M_l(h_i^l, \sum_{j\in\mathcal{N}(i)} AGG_l(h_i^l, h_j^l)) \tag{1}$$

where M_l and AGG_l are update and aggregation functions at the l^{th} layer, respectively and h_i is the embedding at node i. After a sequence of K message passing iterations, a receptive field of size K is covered and the graph embeddings are sent to a readout module to obtain a final prediction.

2.3 Spectral Convolution with Chebyshev Filters

Spectral graph convolutional methods have their foundation in graph signal processing and use graph convolution as aggregating function.[19, 20]. The eigenvectors of the normalized Laplacian form an orthonormal basis in which convolution occurs through simple multiplicative update. The main drawback of spectral methods is that eigendecomposition of the Laplacian can be both computationally expensive and memory intensive for large graphs but it also yields to global updates that do not exploit the potential small support of receptive fields. This issue was addressed and tackled in ChebNet [21] which makes use of a truncated expansion of Chebyshev polynomials to parametrize localized spectral filters as a polynomial function. The filtering operation can be defined as:

$$x * g_\theta = \sum_{k=0}^{K-1} \theta_k T_k(L) x \tag{2}$$

where θ_k is a set of learnable parameters for the K layers, $L = \frac{2L}{\lambda_{max}} - I_n$ and T_k is the K^{th} Chebyshev polynomial.

2.4 Long-range dependencies in graphs

As we consider larger networks with long-range dependencies, GNNs suffer from a major limitation highlighted recently in [5] and referred to as over-squashing whereby information from distant nodes is squashed into fixed-sized vectors as the number of layers grows exponentially. GNNs underperform on this type of graph prediction tasks. In [15], the authors provide a topological perspective on oversquashing through the Balanced Forman curvature measure. They find bottlenecks to be concentrated in regions with high curvature on the edges. Another perspective on oversquashing quantifies the information bottleneck through effective resistance in the graph (Appendix B); similar to electrical networks, a high resistance in a graph impedes the information flow [15, 17, 22]. A spectral metric for oversquashing is given by the Cheeger constant [23, 24] which quantifies the presence of bottlenecks in a graph. This constant is an upper bound for the spectral gap, which corresponds to the smallest non-zero eigenvalue of the Laplacian and describes graph connectivity and expansion properties: information diffuses quicker with a larger spectral gap (high Cheeger constant).

2.5 Graph representation learning for proteins

Graph Neural Networks have become a key component in computational biology due to their ability to represent complex molecular surfaces and learn useful interactions among the atoms in those systems. A notable use-case is protein function prediction. In [25], Gliborijevic et.al present a graph-based architecture that takes as input a protein structure and a sequence from a pre-trained language model. The model predicts the function of the protein and the key residues in the sequence for that function. Another important application is protein structure comparison which is crucial for structural homology discovery and other downstream structure-based analysis. For this purpose, GraSR was introduced in [26] as a graph contrastive model to better learn global and local geometric features of residues. Recent studies on molecular graphs discuss the importance of combining local semantics carrying potentially critical information about graph substructures with graph-level features summarizing its global topology [27, 28].

3 Tackling long-range dependencies in graphs

3.1 Latent learning

The idea of introducing a latent space to constrain the learning task has been explored a few times. In [29], the Perceiver architecture builds on Transformers by injecting latent vectors that aggregate information across different modalities through cross-attention. As a result, the latent vectors technique made the model not only scalable to larger inputs but also generalizable to many modalities [30]. Another advantage of latent space learning can be seen in [31] where non-local representations of images are obtained by fusing their features into a compressed latent space. While there are a few studies on non-local graph neural networks [32], the idea of compressing graph embeddings into smaller non-local representations by means of latent nodes is under-explored, especially for large protein graphs where distant communication is needed. Considering that both scalability and non-locality are desirable properties for modeling protein graphs, it feels natural to fill this gap.

3.2 Graph Rewiring

Given a graph $G = (V, E)$, graph rewiring is the process of changing the graph connectivity through an operation $R(G)$ resulting in a new structure described by $R(G) = (V, R(E))$. The changes mainly include edge addition or deletion

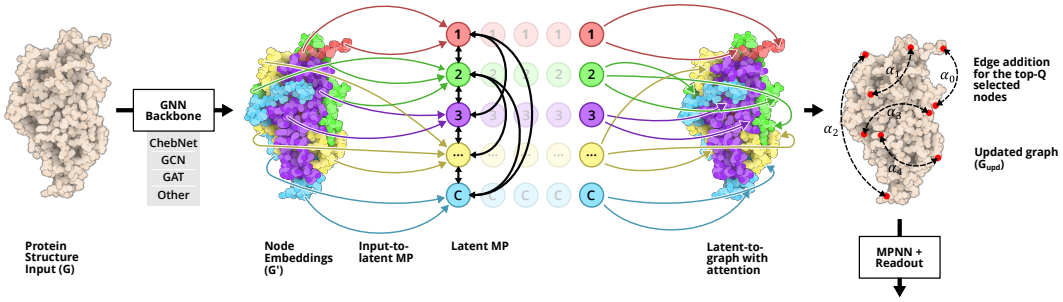

Figure 1: Pipeline of our framework

and the goal is to optimize the information flow within the network to ensure that relevant communication within distant nodes is achieved. Graph rewiring has been frequently addressed in the literature given its diverse set of applications found in traffic management, communication systems and other domains [33, 34]. To tackle oversquashing through rewiring, the authors in [15] consider the graph topology and change the graph connectivity by adding edges in regions of low curvature and disconnecting high-curvature regions as a way of alleviating the bottlenecks. Alternatively, the authors in [35] present EdgeRewire, a rewiring framework that optimizes various spectral constraints of the graphs to make them more robust. Some of the main metrics for bottlenecks include the spectral gap and the effective resistance. These two metrics have been revisited in the recent studies on oversquashing connecting them to long-range dependencies in graphs. For instance, the DiffWire framework in [17] performs rewiring through a GAP layer that optimizes the spectral gap of the network. Analogously, the work in [16] provides a computationally efficient way to add edges based on the spectral gap.

4 Graph neural networks for long-range dependencies in proteins

Consider a protein represented as a graph $G = (X, A)$ whereby each atom/node in $X \in \mathbb{R}^{n \times d}$ is a feature vector $h \in \mathbb{R}^d$. In this section we provide a novel use-case based on attention-based rewiring of protein structures and proceed to explain the choice of a spectral backbone and its ability to boost the performance in graph learning with long-range dependencies. This approach is motivated by the fact that edge addition in general results in an increased spectral gap and hence reduced bottlenecks in the network as described in Section 3.2. Adapting a pragmatic approach on which edges to add is crucial to prevent oversmoothing by adding too many unnecessary connections. A mathematical description of the effect edge addition has on the spectral gap is found in [35].

4.1 A local-to-global latent GNN

In this subsection we summarize the main steps of our architecture shown in Figure 1.

Step 1: Intra-atoms message passing. We begin by covering the local neighborhoods of the protein, so we update the node representations based on their surrounding neighbors through a graph convolution operation Θ_{conv} resulting in new atom embeddings $h'_{local} \in \mathbb{R}^f$.

Step 2: Protein-to-LN message passing. We propose to fuse the local graph information into a compressed global representation through message passing between the local embeddings h'_{local} and a set of c trainable latent nodes forming a graph $G_c \in \mathbb{R}^{c \times f}$ such that $c << n$. Given the embedding $h'_{i-local}$ for each node i in G and the initial latent embedding z_q for a node $q \in [1, c]$, we perform an input-to-latent message passing:

$$z_q^{(l+1)} = M_l(z_q^l, \sum_{j \in \mathcal{N}(q)} AGG_l(z_q^l, h_j^l)) \tag{3}$$

For this message passing to occur, the nodes in G and the latent nodes are considered to be part of a bipartite graph. Each node in G is randomly assigned to a latent node in G_c, hence two or more nodes in G can be connected to the same latent node. A total of $|E| = n$ edges need to be added, alleviating some computational cost in comparison with a Transformer where each node attends all other nodes. Instead, we use a Graph Attention Network (GAT) [36] to aggregate messages towards the latent components.

Step 3: Intra-LN message passing. The latent nodes then exchange information through fully connected message passing i.e each latent node attends all other C nodes as described below:

$$z_q^{(l+1)} = M_l(z_q^l, \sum_{j \in [1,c]} AGG_l(z_q^l, z_j^l)) \tag{4}$$

This step ensures that latent nodes covering different ranges of the protein have interacted, providing a solid global operator for protein representation.

Step 4: Attention-based rewiring through latent nodes. This methodology is based on the idea that distant nodes need to communicate for a more effective representation at the graph-level, hence it uses attention as metric to quantify the importance of the interaction of two distant nodes using the latent nodes as a mediator. In contrast to previous methods where rewiring is based on pre-processed measures [15, 37], we perform rewiring in an end-to-end fashion in analogy to more recent work [17, 38] . To proceed, we use an additional GAT to perform message passing from the updated latent nodes back to the nodes $h_{i-local}$ as described in Eq.5. We then add new edges among the Q nodes with the highest attention scores and perform graph convolution operations on the updated graph with the new adjacency matrix $G_{rew} = (X, A')$ s.t $A' = A + Q_{edges}$ (Eq.6). For the final classification task, we combine the information from the node embeddings h' and h_{rew} in addition to the latent graph G_c. A pseudo-code is given in Algorithm 1.

$$h_{loc}^{(l+1)} = N_l(h_{loc}^l, \sum_{j \in \mathcal{N}(h)} \alpha \times AGG_l(h_{loc}^l, z_j^l)) \tag{5}$$

$$h_{rew} = P_l(h_{loc}^{l+1}, \sum_{j \in \overline{\mathcal{N}}(h)} Top_Q(\alpha) \times AGG_l(h_{loc}^{l+1}, h_j^l)) \tag{6}$$

Readout: The final step consists of feeding the embeddings to a Multi-layer Perceptron to get the readout of the task. We find experimentally that combining the embeddings h_{rew} from the rewired graph with the latent node embeddings scaled by a hyperparameter λ provides the best performance as show in 7

$$Readout = MLP(h_{rew} + \lambda\, z_{final}) \tag{7}$$

with z_{final} being the final aggregation of the latent nodes obtained by taking either the sum or the mean.

Algorithm 1 Attention-based rewiring through latent nodes

Input: Graph $G = (X, A)$
Initialize: $\theta \leftarrow \theta_0, \phi \leftarrow \phi_0, V_c \leftarrow rand(C, F)$
repeat
  $G_{i+1} \leftarrow \text{GNN}_\theta(G_i)$ *Backbone graph convolution*
until i==Z
$V_c \leftarrow G_Z$
$\phi(V_c) \leftarrow V_c$ *Fully-connected latent MP*
$G_Z \leftarrow \phi(V_c); \alpha \in \mathbb{R}^E$ *GAT from latent to input*
$E_{att} \leftarrow max_\alpha^Q$ *Select top-Q attention values*
$E_{upd} \leftarrow concat(E_G, E_{upd})$ *Update adjacency matrix*
repeat
  $G_{j+1} \leftarrow \text{GNN}_\gamma(G_Z)$ *updated GCN*
until j==2

5 Experiments

Protein 3D structures We benchmark on a collection of protein structure datasets obtained using the ProteinShake framework [39]. We first validate our model on the EnzymeClass task which consists of predicting the type of reaction catalyzed by the given protein as given by the Enzyme Commision database. The second task consists of binding site identification on the PDBBind2020 dataset [40]. Both datasets are also publicly available on the Protein Data Bank platform [41].

Peptide dataset We evaluate the performance of our model on the Peptide datasets of the Long Range Graph Benchmark (LRGB) which is the main benchmark for learning on graphs with long-range dependencies. Peptides are short chains of amino acids that play a crucial role in biological processes. They exhibit distant dependencies through the long-range

amino acid interactions that influence the chain. In this scenario, amino acids are mapped as nodes and the whole peptide structure is considered to be a graph. The Peptides dataset is split into two tasks: graph classification (Petptides-func) and graph regression (Peptides-struct).

Structure comparison Inspired by work on graph isomorphism and graph comparison, we evaluate the ability of an MPNN to distinguish graph-level proteins and compare their structures and family. We compare this ability to that of ChebNet which has been theoretically shown to have higher expressivity [42]. The Structural Similarity dataset was generated using the TM-Align software [43] and is available on the PDB Database [41]. Given the scarcity of long-range datasets with challenging substructure comparison, we add a jet tagging task obtained from CERN's open data portal.

Robustness We validate our model's ability to capture long-range node dependencies by testing it on the synthetic RingTransfer dataset presented in [44]. This dataset contains graphs in the form of rings of size k (chordless cycles). Within each graph, we identify two specific nodes as the target and source, consistently positioning them at a distance of $\frac{k}{2}$. Given a source node, the goal is to obtain a hot-one encoding of its label at the level of a target node, with all remaining nodes containing a uniform feature vector. While previous datasets demonstrate the ability of our latent-based model in capturing long-range dependencies, RingTransfer highlights its robustness as the network depth increases.

Baselines We provide a comparison with a set of baselines to highlight the benefit of rewiring through attention in capturing long-range dependencies, in addition to the expressive power of ChebNet as a backbone on these types of graphs. For the Peptide function and structure, we provide a comparison with SOTA architectures such as Drew [38] and other rewiring approaches on that dataset [16, 13]. [1].

6 Results

Results on different protein-level and atom-level tasks are summarized in Tables 1 and 2, respectively. It is shown as expected that adding multiple latent nodes to cover different regions of the proteins boosts the performance relative to using the backbone alone. Further details on training is found in Appendix A. For the PDBBind dataset, we disregard the GCN and Cheb-latent architectures given that as a node classification task, and in contrast to the graph-level ones, we do not directly use the latent nodes in the classifier, but rather only the messages propagating back from them (as in Cheb-Rewire and GCN-Rewire).

Attention-based rewiring advantage: We highlight the improvement obtained by the rewiring framework when using the GCN backbone. On all the benchmark datasets, GCN-Rewire performs better than both the GCN backbone and the GCN model with multiple virtual nodes. On the Peptide dataset, it surpasses other rewiring frameworks such as FOSR and LASER. The advantage is especially shown on the binding site detection task in the PDBBind dataset, which is the only atomic-level task. This can be explained by the success of this method in attending distant residues whose communication potentially determines the overall protein's function. By doing so, it provides a solid combination of local and distant neighboring features on the one hand through both features of h_{local} and h_{rew} and the global features on the other hand through the latent node features z_q. We evaluated the model under different parameters and it is found empirically that the best performance saturates around $K = 6$ latent nodes and $Q = 8$ newly rewired nodes.

ChebNet case: Having shed light on the powerful aspect of rewiring through latent nodes, we now highlight the solid improvements obtained by adapting a pragmatic choice of a backbone which is explained further in Section 6.1. It can be seen empirically that, **without any rewiring or latent nodes**, Chebyshev Convolution (ChebNet) alone outperforms most baselines. It has a similar performance to SOTA models such as Drew on the peptide function prediction, while surpassing it on the structure regression. The rewiring framework on top of ChebNet still helps boost the performance in the Enzymes and PDBBind datasets, which can be explained by their relatively larger size. On those datasets, the Transformer and ChebNet variants show a more notable advantage. In a third set of experiments, we experimentally validate on proteins the expressive power of ChebNet relative to modern MPNNs that was theoretically discussed in [45, 42]. As shown in Table 3, ChebNet notably surpasses an MPNN on a graph-level scenario for distinguishing two large graphs. This can attributed to its ability to learn global and local structures simultaneously. The latter make the usage of a spectral backbone like ChebNet rely capable of capturing non-localities, **hence virtual nodes and rewiring show less competitive advantages on top of ChebNet.**

6.1 The expressive power of a spectral backbone: ChebNet

Multi-hop receptive field: One major factor that enables the ChebNet architecture to perform better is its K-hop localized nature. Eq.2 describing ChebNet is a polynomial function parametrizing spectral filters, the latter is taking

[1]The code is made available at: https://anonymous.4open.science/r/Protein-Structures-2582/.

Model	Peptide-function	Peptide-structure
	Test AP ↑	Test MAE ↓
GCN	0.5930 ± 0.0023	0.3496 ± 0.0013
GCN-Latent	0.6211 ± 0.0059	0.2723 ± 0.0040
GCN-Rewire	**0.6670 ± 0.0024**	**0.2660 ± 0.0043**
GAT	0.5800 ± 0.0061	0.3506 ± 0.0011
FOSR	0.4629 ± 0.0071	0.3078 ± 0.0026
LASER	0.6447 ± 0.0033	0.3151 ± 0.0006
Transformer + PE	0.6326 ± 0.0126	**0.2529 ± 0.0016**
Drew-GCN	**0.6996 ± 0.0076**	**0.2781 ± 0.0028**
ChebNet	**0.6946 ± 0.0044**	**0.2583 ± 0.0021**
ChebNet-Latent	**0.6820 ± 0.0131**	**0.2582 ± 0.0011**
ChebNet-Rewire	**0.6766 ± 0.0235**	**0.2599 ± 0.0029**

Table 1: We compare the performance of latent-based models on the LRGB datasets against numerous baselines. We colour based on the ranking: **first** , **second** and **third**.

Model	EnzymesClass	PDBBind
Type	Graph-level	Node-level
	Test Acc ↑	Test AUROC ↑
GCN	73.33 ± 1.06	62.65 ± 0.13
GCN-Latent	74.22 ± 0.50	N/A
GCN-Rewire	**75.80 ± 0.92**	**66.53 ± 0.34**
ChebNet	**76.57 ± 1.51**	**71.70 ± 0.23**
ChebNet-Latent	**76.89 ± 0.64**	N/A
ChebNet-Rewire	**77.87 ± 0.86**	**73.37 ± 0.44**

Table 2: We compare the performance of latent-based models on protein structure datasets against numerous baselines.

Model	Protein Structural Similarity	Quark-Gluon decay
	Test Spearman Corr. ↑	Test Acc ↑
MPNN	56.79 ± 1.46	61.90 ± 1.20
ChebNet	**63.25 ± 1.44**	**65.36 ± 1.34**

Table 3: Performance comparison on protein graph comparison between MPNN and ChebNet

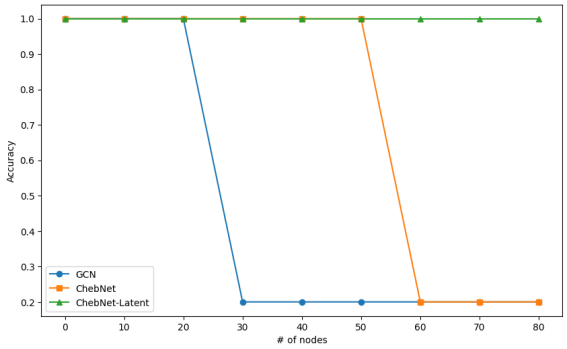

Figure 2: Performance comparison on the RingTransfer dataset

powers of the Laplacian and weighting them by the parameter θ_k prior to obtaining the new node representation. By doing so, the spectral filter gradually aggregates information from a receptive field of size $k \in [0, K]$ for each k^{th} term of the polynomial. Therefore, ChebNet can be seen as a multi-hop graph neural network as it accounts for a receptive field of size K prior to a node embedding. This strategy has been shown to work better on datasets with long-range dependencies in the Drew and LASER methods [13, 38] as it dampens the loss of information between distant nodes and hence prevents oversquashing. In addition, the weighting term θ_k is useful to assign more importance to distant hops when needed: this is especially useful on datasets with high-frequency signals.

Robustness to growing receptive field: Figure 2 shows a comparison of the performance of ChebNet on the Ring-Transfer dataset in its backbone and latent model form against a simple GCN. The GCN can perform well up until 30 nodes, above which the performance drops to 0.2. As the diameter of the ring increases, the number of layers required to reach the target node is higher, resulting in oversmoothing. This is not the case for ChebNet which can perform consistently well up to 55 nodes in its simplest form prior to failure, in contrast to the latent ChebNet model which is consistently robust to the increase in diameter. We analyze the potential reasons for the improvement provided by the choice of ChebNet in the next section.

Jacobian bound: To analyze over-squashing, the authors in [15] provide a formulation of the Jacobian of the embedding h_i^r of a node i relative to the initial features h_j^0 of a node j at distance r. The Jacobian is an important measure to quantify the information flow between nodes i and j after r message passing iterations. More specifically, a low Jacobian value indicates that h_j^0 has a minimal effect on h_i^r due to the *squashing* of information into fixed-size vectors. The interaction of these nodes is bounded by the powers of the adjacency matrix A through the following formulation:

$$\left| \frac{\partial h_i^r}{\partial h_j^0} \right| \leq c(A^r)_{ij} \tag{8}$$

In our case using ChebNet, we use the normalized Laplacian adjacency matrix defined in Section 2.1. ChebNet spans k-hop layers per iteration and hence information between two nodes at distance r is reached in $\lfloor \frac{r}{k} \rfloor$ iterations. Consequently, the exponential decay proportional to r given in Eq.6 becomes slower as the Jacobian is now dependent on $\hat{A}^{\frac{r}{k}}$ where $\frac{r}{k} << r$. By increasing this upper bound on the Jacobian, the latter now has a larger value which translates into an alleviated over-squashing, hence explaining the better performance ChebNet provides on datasets displaying long-range dependencies.

Link to Commute Time: As described above, ChebNet requires $\lfloor \frac{r}{k} \rfloor$ iterations to go from a node i to a node j at a distance r. Consequently, the commute time between these nodes is also reduced from r to $\lfloor \frac{r}{k} \rfloor$ as the shortest paths are considered. Commute-time and Effective Resistance (which are often used interchangeably) have been directly linked to over-squashing. The latter becomes more prevalent in graph tasks which depend on the interactions between nodes with high CT due to high obstruction during information transfer. We refer the reader to **Appendix** in [24] for a detailed description of this connection.

Alleviating Oversmoothing: A starting point is to observe the spectral backbone from the perspective of the influence score in analogy to [46]. Given a starting node A and a distant node B in a graph $G = (V, E)$ the influence score $I(A, B)$ of node A by B is the sum of the absolute values of the entries of the Jacobian matrix $[\frac{\partial h_A^{(k)}}{\partial h_B^{(0)}}]$. In the case of ChebNet, if we consider the case at the 0^{th} layer, this ratio is proportional to the parameter θ_0^r at the r^{th} neighborhood where B falls, i.e $[\frac{\partial h_A^{(0)}}{\partial h_B^{(0)}}] \propto \theta_0^r$. After k convolutional layers, the embedding h'_A of A is proportional to the embedding h'_B of B through a new parameter θ_k^r while still inherently containing information from θ_0^r.Hence, the inherent information from earlier convolutional layers can be seen as a sort of residual connections from the initial layer to later ones. Hence the strong performance from ChebNet is logical considering that residual connections are one way to tackle dependencies and prevent oversmoothing [46].

7 Conclusion

In this work, we have proposed a unified framework to alleviate the phenomenon of oversquashing that GNNs exhibit on proteins when dealing with long-range dependencies. The main components of the framework include the use of latent nodes that cover different regions of proteins and a novel use-case these latent nodes as a mediator for rewiring. We show that latent nodes can enhance the performance on a given GNN backbone for the datasets under consideration. We also evaluate our extension on the same dataset and show the additional boost it provides. We study the design space of possible GNN backbones and find empirically that the spectral ChebNet model outperforms other baselines. We attribute this superiority to its weighted multi-hop aspect and provide an analysis of this result. Future work can make use of this methodology to further extend its expressive power by trying more pragmatic approaches to define the input to latent connections. Another perspective can be to view the method as a set learning problem as described in [47]. It could also be interesting to explore the latent space exhibited by the latent nodes for interpretability.

References

[1] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[3] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24), 2018.

[4] Johannes Gasteiger, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

[5] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

[6] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in {gnn}s. In *International Conference on Learning Representations*, 2020.

[7] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

[8] Chen Cai and Yusu Wang. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

[9] M Michael Gromiha and Samuel Selvaraj. Importance of long-range interactions in protein folding. *Biophysical chemistry*, 77(1):49–68, 1999.

[10] Daisuke Kihara. The effect of long-range interactions on the secondary structure formation of proteins. *Protein Science*, 14(8):1955–1963, 2005.

[11] M Michael Gromiha and S Selvaraj. Comparison between long-range interactions and contact order in determining the folding rate of two-state proteins: application of long-range order to folding rate prediction. *Journal of molecular biology*, 310(1):27–32, 2001.

[12] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.

[13] Federico Barbero, Ameya Velingker, Amin Saberi, Michael Bronstein, and Francesco Di Giovanni. Locality-aware graph-rewiring in gnns. *arXiv preprint arXiv:2310.01668*, 2023.

[14] Chen Cai, Truong Son Hy, Rose Yu, and Yusu Wang. On the connection between mpnn and graph transformer. *International Conference on Machine Learning*, 2023.

[15] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

[16] Kedar Karhadkar, Pradeep Kr. Banerjee, and Guido Montúfar. Fosr: First-order spectral rewiring for addressing oversquashing in gnns. In *The Eleventh International Conference on Learning Representations, ICLR*, 2023.

[17] Adrián Arnaiz-Rodríguez, Ahmed Begga, Francisco Escolano, and Nuria M Oliver. DiffWire: Inductive Graph Rewiring via the Lovász Bound. In Bastian Rieck and Razvan Pascanu, editors, *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pages 15:1–15:27. PMLR, 09–12 Dec 2022.

[18] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[19] Antonio Ortega, Pascal Frossard, Jelena Kovačević, José MF Moura, and Pierre Vandergheynst. Graph signal processing: Overview, challenges, and applications. *Proceedings of the IEEE*, 106(5):808–828, 2018.

[20] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[21] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[22] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in gnns through the lens of effective resistance. In *International Conference on Machine Learning*, pages 2528–2547. PMLR, 2023.

[23] Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

[24] Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Lio, and Michael M Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *International Conference on Machine Learning*, pages 7865–7885. PMLR, 2023.

[25] Vladimir Gligorijević, P Douglas Renfrew, Tomasz Kosciolek, Julia Koehler Leman, Daniel Berenberg, Tommi Vatanen, Chris Chandler, Bryn C Taylor, Ian M Fisk, Hera Vlamakis, et al. Structure-based protein function prediction using graph convolutional networks. *Nature communications*, 12(1):3168, 2021.

[26] Chunqiu Xia, Shi-Hao Feng, Ying Xia, Xiaoyong Pan, and Hong-Bin Shen. Fast protein structure comparison through effective representation learning with contrastive graph neural networks. *PLoS computational biology*, 18(3):e1009986, 2022.

[27] Hanchen Wang, Jean Kaddour, Shengchao Liu, Jian Tang, Joan Lasenby, and Qi Liu. Evaluating self-supervised learning for molecular graph embeddings. *arXiv preprint arXiv:2206.08005*, 2022.

[28] Xiao Luo, Wei Ju, Meng Qu, Yiyang Gu, Chong Chen, Minghua Deng, Xian-Sheng Hua, and Ming Zhang. Clear: Cluster-enhanced contrast for self-supervised graph representation learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[29] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[30] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

[31] Songyang Zhang, Xuming He, and Shipeng Yan. Latentgnn: Learning efficient non-local relations for visual recognition. In *International Conference on Machine Learning*, pages 7374–7383. PMLR, 2019.

[32] Meng Liu, Zhengyang Wang, and Shuiwang Ji. Non-local graph neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 44(12):10270–10276, 2021.

[33] Sarah Dunn and Sean M Wilkinson. Increasing the resilience of air traffic networks using a network graph theory approach. *Transportation Research Part E: Logistics and Transportation Review*, 90:39–50, 2016.

[34] John S Baras and Pedram Hovareshti. Efficient and robust communication topologies for distributed decision making in networked systems. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3751–3756. IEEE, 2009.

[35] Hau Chan and Leman Akoglu. Optimizing network robustness by edge rewiring: a general framework. *Data Mining and Knowledge Discovery*, 30:1395–1425, 2016.

[36] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018.

[37] Jhony H Giraldo, Konstantinos Skianis, Thierry Bouwmans, and Fragkiskos D Malliaros. On the trade-off between over-smoothing and over-squashing in deep graph neural networks. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 566–576, 2023.

[38] Benjamin Gutteridge, Xiaowen Dong, Michael M Bronstein, and Francesco Di Giovanni. Drew: Dynamically rewired message passing with delay. In *International Conference on Machine Learning*, pages 12252–12267. PMLR, 2023.

[39] Tim Kucera, Carlos Oliver, Dexiong Chen, and Karsten Borgwardt. Proteinshake: Building datasets and benchmarks for deep learning on protein structures. *Advances in Neural Information Processing Systems*, 36, 2024.

[40] Renxiao Wang, Xueliang Fang, Yipin Lu, and Shaomeng Wang. The pdbbind database: Collection of binding affinities for protein- ligand complexes with known three-dimensional structures. *Journal of medicinal chemistry*, 47(12):2977–2980, 2004.

[41] Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[42] Muhammet Balcilar, Pierre Héroux, Benoit Gauzere, Pascal Vasseur, Sébastien Adam, and Paul Honeine. Breaking the limits of message passing graph neural networks. In *International Conference on Machine Learning*, pages 599–608. PMLR, 2021.

[43] Yang Zhang and Jeffrey Skolnick. Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic acids research*, 33(7):2302–2309, 2005.

[44] Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in Neural Information Processing Systems*, 34:2625–2640, 2021.

[45] Muhammet Balcilar, Renton Guillaume, Pierre Héroux, Benoit Gaüzère, Sébastien Adam, and Paul Honeine. Analyzing the expressive power of graph neural networks in a spectral perspective. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[46] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018.

[47] Konstantinos Skianis, Giannis Nikolentzos, Stratis Limnios, and Michalis Vazirgiannis. Rep the set: Neural networks for learning set representations. In *International conference on artificial intelligence and statistics*, pages 1410–1420. PMLR, 2020.

A Dataset Summary

We report the properties of different datasets in the Table below. For training on all datasets, we used the AdamW optimizer with the learning rates tabulated below. All experiments are done on an NVIDIA Titan-RTX GPU and we report an average of 3 runs.

Dataset	# of graphs	Loss Function	hidden dim	Learning rate
Peptide-func	15,535	Cross-Entropy	300	0.0001
Peptide-struc	15,535	MSE	300	0.0001
PDBBind	2839	Cross-Entropy	150	0.0005
EnzymesClass	15,603	Cross-Entropy	150	0.0005
Structural Similarity	994	MSE	150	0.0005

Table 4: Dataset description

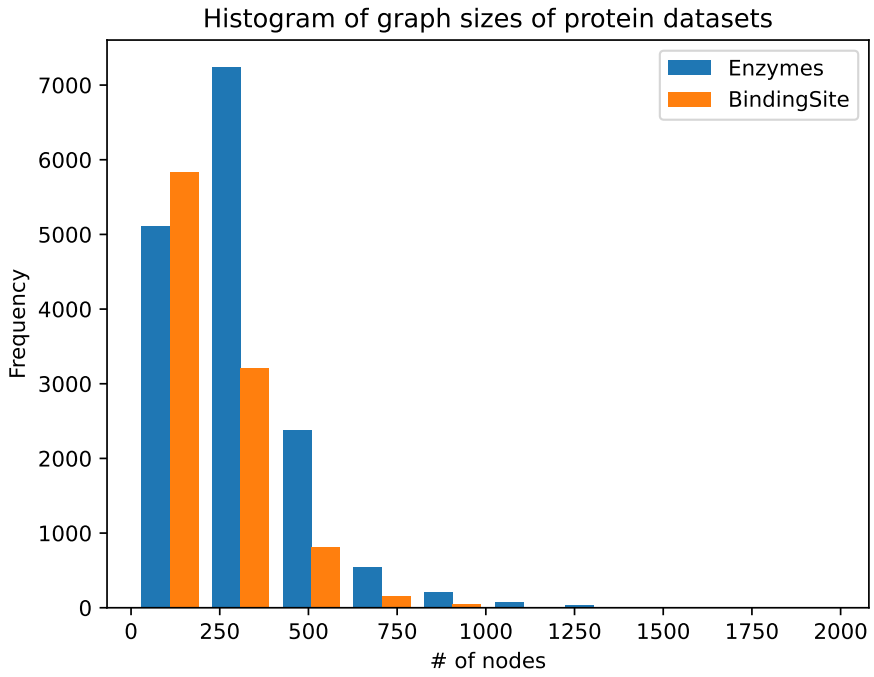

Figure 3: Number of nodes across different ProteinShake Datasets

B Definitions

Effective resistance (ER) in a graph measures the resistance between two nodes when an electrical current is passed through the edges. It quantifies how well-connected or isolated the nodes are and is commonly used in network analysis to assess the flow of information or current within the graph.

Commute time (CT) in a graph represents the expected time it takes for a random walk or particle to travel between two nodes, starting from one and reaching the other. It is a measure of the efficiency of traversal within the graph and finds applications in various fields, such as computer science and transportation planning.