PANDA: Expanded Width-Aware Message Passing Beyond Rewiring

Jeongwhan Choi [1] **Sumin Park** [2] **Hyowon Wi** [1] **Sung-Bae Cho** [1] **Noseong Park** [3]

Abstract

Recent research in the field of graph neural network (GNN) has identified a critical issue known as "over-squashing," resulting from the bottleneck phenomenon in graph structures, which impedes the propagation of long-range information. Prior works have proposed a variety of graph rewiring concepts that aim at optimizing the spatial or spectral properties of graphs to promote the signal propagation. However, such approaches inevitably deteriorate the original graph topology, which may lead to a distortion of information flow. To address this, we introduce an ex**pand**ed width-**a**ware (**PANDA**) message passing, a new message passing paradigm where nodes with high centrality, a potential source of over-squashing, are selectively expanded in width to encapsulate the growing influx of signals from distant nodes. Experimental results show that our method outperforms existing rewiring methods, suggesting that selectively expanding the hidden state of nodes can be a compelling alternative to graph rewiring for addressing the over-squashing.

1. Introduction

Graph Neural Networks (GNNs) have emerged as a powerful tool for graph data processing and are being studied extensively in various domains, as evidenced by significant research contributions (Defferrard et al., 2016; Veličković et al., 2018; Chen et al., 2020; Chien et al., 2021; Choi et al., 2021; Chamberlain et al., 2021; Hwang et al., 2021; Choi & Park, 2023; Choi et al., 2023; Kang et al., 2023; Gruber et al., 2023). A key subclass within GNNs is Message Passing Neural Networks (MPNNs), which excel in propagating information through neighbouring nodes. As MPNNs propagate only for a 1-hop distance within a single layer, long-range interactions can only be captured when the depth

[1]Yonsei University, Seoul, South Korea. [2]DNI Consulting, Seoul, South Korea. [3]KAIST, Daejeon, South Korea. Correspondence to: Noseong Park <noseong@kaist.ac.kr>.
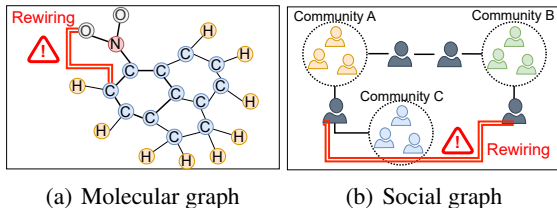
Figure 1. Potential pitfalls of rewiring in domain-specific graphs: (a) In a molecular graph, rewiring the edge in red to a benzene ring violates the domain knowledge. (b) In a social graph, connecting a user to his/her enemy may lead to totally different meaning.
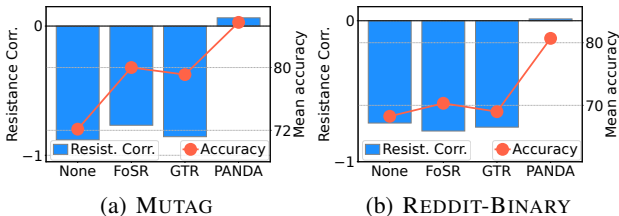

Figure 2. Comparison of resistance correlation and mean accuracy across different methods for GCN. A large negative correlation reflects that a higher total effective resistance is associated with reduced signal propagation (see Section 4).

of the network is comparable to the distance. However, increasing the number of layers for capturing long-range dependencies results in exponential growths of the receptive field and excessive aggregations of repeated messages that are subsequently packed into a feature vector of fixed length, which causes the problem called "over-squashing" (Alon & Yahav, 2021; Shi et al., 2023b). This phenomenon severely decreases the expressivity of the networks and thus impairs the model performance. Previous studies have attempted to identify topological bottlenecks in a graph and directly modify graph connectivity with various optimization targets, including curvature, effective resistance, and spectral gap (Alon & Yahav, 2021; Topping et al., 2022; Banerjee et al., 2022; Deac et al., 2022; Arnaiz-Rodríguez et al., 2022; Karhadkar et al., 2023; Black et al., 2023; Fesser & Weber, 2023; Sonthalia et al., 2023; Giraldo et al., 2023; Shi et al., 2023a; Barbero et al., 2023).

Motivation. Existing rewiring methods that alter the graph topology to resolve over-squashing, while potentially beneficial, can inadvertently introduce inaccuracies within domain-specific contexts. Fig. 1(a) demonstrates how modifying the molecular graph of a benzene ring could contradict chemical principles, while Fig. 1(b) shows that rewiring in social networks could result in the distortion of underlying community structures. These examples highlight the necessity to preserve the original graph structure to maintain the validity of domain-specific semantic information.

Prior works have tried to identify various factors associated with the over-squashing through the lens of sensitivity analysis of Jacobian of node features (Topping et al., 2022; Di Giovanni et al., 2023). As one of those trials, Di Giovanni et al. (2023), an insightful analytical paper, provides a theoretical justification that increasing the width of the model (i.e., hidden dimension) can also contribute to improving the sensitivity of the model. However, it points out a major concern regarding expanding the width of the model to address over-squashing; increasing the hidden dimension globally affects the whole networks at the expense of the generalization capacity of the model.

Inspired by the limitation of existing rewiring methods and the promising role of width expansion for addressing the over-squashing, we aim to design a novel message passing paradigm that mitigates the over-squashing by targeting the nodes that are mainly involved in creating bottlenecks in the signal flow and selectively expanding their width without the risk to change the underlying graph topology.

Main idea. It is possible to define graph bottlenecks as nodes with high centrality, such as betweenness centrality (Yu et al., 2007; Topping et al., 2022). Nodes with high centrality can show hub-like behavior in that they connect to a relatively larger set of nodes lying on the different parts of the graph, and thus receive excessive information from them. Increasing the width of the feature vector of these nodes provides more space for them to process information flowing from other nodes. Then, we can take advantage of their enhanced capacities to alleviate over-squashing. In order to do this, it requires exchanging messages between nodes with different hidden dimensions. This raises a natural question:

"Is it possible to design a message passing that enables information exchange between nodes with different width only in a way that mitigates over-squashing without rewiring?"

We propose a new message passing framework that addresses the above question. First, the entire node set is divided into two subsets, expanded and non-expanded nodes, which are colored in red (⬤) and blue (⬤), respectively in Fig. 3. Then, we classify four different edge types depending on the expansion state of the nodes at source (edge tail) and target (edge head) position: low-to-low, high-to-high,

low-to-high, and high-to-low. Each edge type requires a distinct message passing scheme.

The standard message passing between nodes with equal widths corresponds to the conventional message passing (see Figs. 3(b) and 3(c)). For high-to-low message passing (see Fig. 3(d)), a low-dimensional node selects and receives a subset of the information sent by a high-dimensional node. The low-to-high message passing involves augmenting the hidden vectors of low-dimensional nodes to match the higher dimension size for propagation (see Fig. 3(e)).

Our framework can potentially enhance the capacity of nodes in the perspective of signal propagation. As shown in Fig. 2, our framework maintains constant signal propagation w.r.t. effective resistance and *simultaneously* improves the accuracy compared to the existing rewiring methods. We use the correlation coefficient to quantify how signal propagation decreases as effective resistance increases.

Contributions and outline. We introduce an ex**pand**ed width-**a**ware message passing (**PANDA**)[1], a novel paradigm for message passing with expanded widths for nodes that are potentially bottlenecks. Our contributions can be summarized as follows:

- In Section 3, we propose **PANDA** that enables the signal propagation across the nodes of different widths.

- In Section 4, we discuss how our **PANDA** can alleviate over-squashing in terms of various perspectives. Verifying a higher feature sensitivity compared to other models, we show **PANDA** can overcome topological bottleneck, maintaining consistent signal propagation even under large effective resistance. Finally, we show that our method solves the limitation of existing rewiring methods, e.g., over-smoothing.

- In Section 5, we empirically demonstrate that our **PANDA** outperforms existing rewiring methods.

2. Preliminaries

We first examine the notation used in this paper. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we use $\mathcal{V}$ and $\mathcal{E}$ to denote its nodes and edges, respectively. The nodes are indexed by v and u such that $v, u \in \mathcal{V}$, and an edge connecting nodes v and u is denoted by $(v, u) \in \mathcal{E}$. The connectivity is encoded in the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ where n is the number of nodes. p denotes the width (hidden dimension size), while ℓ is the number of layers. The feature of node v at layer ℓ is written as $\mathbf{h}_v^{(\ell+1)}$.

---

[1] Our source code is available here: https://github.com/jeongwhanchoi/panda.

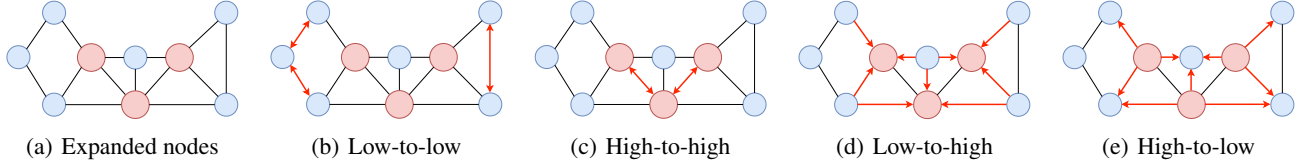| (a) Expanded nodes | (b) Low-to-low | (c) High-to-high | (d) Low-to-high | (e) High-to-low |

Figure 3. Examples of PANDA's message passing mechanism. The size of the node indicates the size of hidden dimension.

2.1. Message Passing Neural Networks

We consider the case where each node v has a feature $\mathbf{h}_v^{(0)} \in \mathbb{R}^p$. MPNNs iteratively update node representations using the following equations:

$$\mathbf{m}_v^{(\ell)} = \psi^{(\ell)}(\{\mathbf{h}_u^{(\ell)} : u \in \mathcal{N}(v)\}), \qquad (1)$$

$$\mathbf{h}_v^{(\ell+1)} = \phi^{(\ell)}(\mathbf{h}_v^{(\ell)}, \mathbf{m}_v^{(\ell)}). \qquad (2)$$

where $\mathbf{m}_v^{(\ell)}$ is the aggregated node feature of v's neighbourhood. The aggregation function $\psi^{(\ell)}$ and the update function $\phi^{(\ell)}$ are learnable, and their different definitions result in different architectures (Kipf & Welling, 2017; Xu et al., 2019; Veličković et al., 2018). In default, all nodes have the same hidden dimension size. We will introduce a method that enables message passing among nodes with different hidden dimensions in Section 3.

2.2. Over-squashing Problem

Initially identified by Alon & Yahav (2021), over-squashing has become a significant challenge in GNNs when dealing with long-range dependencies. It mainly occurs when the information aggregated from too many neighbours is squashed into a fixed-sized node feature vector, resulting in a substantial loss of information (Shi et al., 2023b). We will review the literature, including rewiring methods, to address this problem, in Section 6. We now describe the Jacobian sensitivity and effective resistance widely used to estimate the over-squashing. In Section 4, we justify our method in terms of the two metrics.

Sensitivity bound. Recent theoretical insights on GNNs have provided a formal understanding of over-squashing via the lens of the sensitivity analysis (Topping et al., 2022). The sensitivity bound theorem, as proposed by (Di Giovanni et al., 2023, Theorm 3.2), posits that the impact of over-squashing can be quantified by examining the Lipschitz continuity of the non-linearity in the following 3 factors: i) a model, ii) the width of the model, and iii) the topology of a graph. The sensitivity of $\mathbf{h}_v^{(\ell)}$ to $\mathbf{h}_u^{(0)}$, assuming that there exists a ℓ-path between nodes v and u, is bounded by

$$\left\| \frac{\partial \mathbf{h}_v^{(\ell)}}{\partial \mathbf{h}_u^{(0)}} \right\|_1 \leq \underbrace{(zwp)^\ell}_{\text{model}} \underbrace{(\mathbf{S}^\ell)_{vu}}_{\text{topology}}. \qquad (3)$$

The impact of the model to over-squashing is bounded by the Lipschitz constant z, the width p, and the maximum weight w, while the topology of an input graph affects the sensitivity by ℓ-th power of the graph shift matrix $\mathbf{S}$. Hereinafter, $\mathbf{S}$ can be any (normalized) adjacency matrix (with self-loops).

A small Jacobian norm indicates that the final representations of a node learned by a model are not much affected by the variations in its neighbouring inputs, implying that the network is suffered from over-squashing. This condition typically arises when the information has not been sufficiently propagated over the graph due to the excessive aggregation of information, i.e., compression into a small hidden vector, through several layers of message passing.

Effective resistance and signal propagation. Derived from the field of electrical engineering, the effective resistance between two nodes u and v in an electrical network is defined as the potential difference induced across the edges when a unit current is injected at one of each end (Ghosh et al., 2008). An algebraic expression for the effective resistance is given in Appendix A. Rayleigh's monotonicity principle, which says that adding paths or shortening existing paths can only decrease the effective resistance between two nodes (Thomassen, 1990), leads to the following interpretation: more and shorter disjoint paths connecting the nodes u and v lead to a lower resistance between them (Black et al., 2023; Devriendt & Lambiotte, 2022). Therefore, edges with high effective resistance struggle to propagate information, making bottlenecks. The total effective resistance R_{tot}, the sum of the effective resistance among all pairs of nodes (see Eq. (20)), is a key measure for measuring the overall degree of over-squashing in a graph. It is known that the signal propagation of GNNs is inversely proportional to R_{tot} (Di Giovanni et al., 2023), which will be covered in detail in Section 4 with a comparison to our method.

Over-smoothing problem. Over-smoothing is another well-known problem that reduces the expressiveness of GNNs in deeper layers, resulting in a loss of discriminative power in the representation of node features (Li et al., 2018; Nt & Maehara, 2019; Oono & Suzuki, 2020). Rewiring methods to address the over-squashing problem focus on improving graph connectivity to ease the signal flow. How-

ever, one limitation of the rewiring method is that adding too many edges potentially leads to the over-smoothing issue (Karhadkar et al., 2023). To explore this trade-off between over-squashing and over-smoothing, FoSR (Karhadkar et al., 2023) analyzes the change in Dirichlet energy upon adding edges. Their results demonstrate that rewiring the graph to enhance information propagation leads to the over-smoothing problem.

3. Proposed Method

We first introduce our motivation and design rationale. We then propose our PANDA message-passing framework.

3.1. Motivation and Design Rationale

Sensitivity bound. Our design rationale is anchored in the insight derived from the sensitivity bound theorem (Di Giovanni et al., 2023, Theorem 3.2). Given the sensitivity bound in Eq. (3), we consider two scenarios: one with a standard width p and the other with an increased width p_{high}, where $p_{\text{high}} > p$. For nodes with the increased width, the sensitivity bound is improved, demonstrating a potentially higher sensitivity to input features. Regarding this, we provide a Proposition B.1 in Appendix B and show empirical results well aligned with our argument in Section 4. We do not increase the width of all nodes, but *selectively*.

Design rationale for selecting node centrality. We consider node centrality metrics to identify a node as a potential source of the bottleneck in a graph. Betweenness centrality has been considered as a significant indicator of the bottleneck, which measures the number of shortest paths going through a node (Freeman, 1977). The more frequently a node appears on the shortest paths of pairs of different nodes, the more likely it is to cause the bottleneck (Topping et al., 2022, Definition 9.). However, a hub-like node with inter-community edges also has eccentric properties that are different from those of the high betweenness centrality node, such as a high degree. We apply various centrality metrics in our framework to understand how each is related to the over-squashing problem. Appendix C describes the centralities we consider.

3.2. Expanded Width-Aware Message Passing

Node expansion criteria. In our proposed method, we stratify nodes into two categories: low-dimensional nodes (○) and high-dimensional nodes (○) (see Fig. 4). This stratification is based on a specific centrality that quantifies the importance of each node. The centrality, denoted as $C(\mathcal{G})$, can be one of the following: degree (Borgatti & Halgin, 2011), betweenness (Freeman, 1977), closeness (Freeman, 1978), PageRank (Page et al., 1999), or load centrality (Goh et al., 2001). We represent the centrality values for all nodes
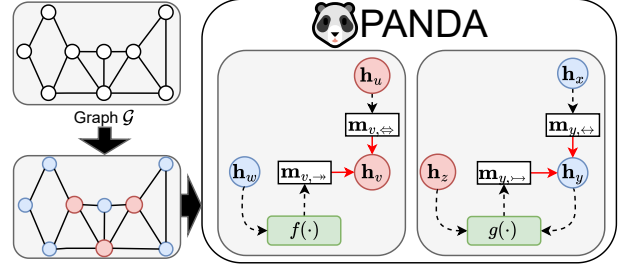

Figure 4. Our proposed **PANDA** message passing framework. First, we selectively expand widths (i.e., hidden dimension sizes) according to a centrality in $\mathcal{G}$, and our **PANDA** message passing enables signal propagation among nodes with different widths (○ low-dimensional nodes and ○ high-dimensional nodes).

as a vector $\mathbf{c} \in \mathbb{R}^n$ and then sort the elements of $\mathbf{c}$ in descending order. Based on the sorted $\mathbf{c}$, we select the top-k nodes, resulting in a binary mask vector, $\mathbf{b} \in \{0, 1\}^n$:

$$\mathbf{b}_v = \begin{cases} 1 & \text{if } v \text{ is among the top } k \text{ nodes in } \mathbf{c}, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Define subsets of neighbourhoods. Based on a centrality measure, we define four subsets of neighbouring nodes. For a node v classified as low-dimensional, denoted by $\mathbf{b}_v = 0$:

$$\mathcal{N}_{\text{low}\leftrightarrow\text{low}}(v) = \{u \in \mathcal{N}(v) \mid \mathbf{b}_v = 0 \wedge \mathbf{b}_u = 0\}, \quad (5)$$

$$\mathcal{N}_{\text{high}\rightarrowtail\text{low}}(v) = \{u \in \mathcal{N}(v) \mid \mathbf{b}_v = 0 \wedge \mathbf{b}_u = 1\}. \quad (6)$$

In addition, for a node v classified as high-dimensional:

$$\mathcal{N}_{\text{high}\Leftrightarrow\text{high}}(v) = \{u \in \mathcal{N}(v) \mid \mathbf{b}_v = 1 \wedge \mathbf{b}_u = 1\}, \quad (7)$$

$$\mathcal{N}_{\text{low}\rightarrow\text{high}}(v) = \{u \in \mathcal{N}(v) \mid \mathbf{b}_v = 1 \wedge \mathbf{b}_u = 0\}. \quad (8)$$

The comprehensive union of these subsets across all nodes in $\mathcal{V}$ encompasses the entire edge set $\mathcal{E}$, thereby preserving the original graph structure.

A new framework: PANDA message passing. The main contribution of our framework is that, given any MPNNs, we can make it work with different widths for nodes. To this end, we replace the standard neighbour aggregator $\psi^{(\ell)}$ with 4 different aggregators, each applied to the corresponding type of neighbours: $\psi_{\leftrightarrow}^{(\ell)}$, $\psi_{\Leftrightarrow}^{(\ell)}$, $\psi_{\rightarrow}^{(\ell)}$, and $\psi_{\rightarrowtail}^{(\ell)}$. Then, we use a common update function for all types. The final message passed from layer ℓ to layer $\ell + 1$ is defined as follows:

$$\mathbf{m}_{v,\leftrightarrow}^{(\ell)} = \psi_{\leftrightarrow}^{(\ell)}(\{\mathbf{h}_u^{(\ell)} : u \in \mathcal{N}_{\text{low}\leftrightarrow\text{low}}(v)\}), \quad (9)$$

$$\mathbf{m}_{v,\Leftrightarrow}^{(\ell)} = \psi_{\Leftrightarrow}^{(\ell)}(\{\mathbf{h}_u^{(\ell)} : u \in \mathcal{N}_{\text{high}\Leftrightarrow\text{high}}(v)\}), \quad (10)$$

$$\mathbf{m}_{v,\rightarrow}^{(\ell)} = \psi_{\rightarrow}^{(\ell)}(\{f(\mathbf{h}_u^{(\ell)}) : u \in \mathcal{N}_{\text{low}\rightarrow\text{high}}(v)\}), \quad (11)$$

$$\mathbf{m}_{v,\rightarrowtail}^{(\ell)} = \psi_{\rightarrowtail}^{(\ell)}(\{g(\mathbf{h}_v^{(\ell)}, \mathbf{h}_u^{(\ell)}) : u \in \mathcal{N}_{\text{high}\rightarrowtail\text{low}}(v)\}), \quad (12)$$

$$\mathbf{h}_v^{(\ell+1)} = \phi(\mathbf{h}_v^{(\ell)}, \mathbf{m}_{v,\leftrightarrow}^{(\ell)}, \mathbf{m}_{v,\Leftrightarrow}^{(\ell)}, \mathbf{m}_{v,\rightarrow}^{(\ell)}, \mathbf{m}_{v,\rightarrowtail}^{(\ell)}). \quad (13)$$

Each aggregator $\psi_*^{(\ell)}$ is tailored to handle a specific type of interactions among nodes. In Eq. (13), the hidden vector of a node v is updated from the four messages. For instance, $\psi_{\twoheadrightarrow}^{(\ell)}(\cdot)$ and $\psi_{\rightarrowtail}^{(\ell)}(\cdot)$ are designed for interactions with different widths by using functions $f(\cdot)$ and $g(\cdot)$.

$f(\cdot)$ is a linear transformation that projects node features to the expanded dimensional space and is defined as:

$$f(\mathbf{h}_u^{(\ell)}) := \mathbf{W}_f^{(\ell)} \mathbf{h}_u^{(\ell)}, \qquad (14)$$

where $\mathbf{W}_f^{(\ell)} \in \mathbb{R}^{p_{\text{high}} \times p}$. For $\psi_{\rightarrowtail}^{(\ell)}(\cdot)$, we define $g(\cdot)$ as a dimension selector as follows:

$$g(\mathbf{h}_v^{(\ell)}, \mathbf{h}_u^{(\ell)}) := \text{gather}\left(\mathbf{h}_u^{(\ell)}, \text{topk}(\mathbf{s}, p)\right), \qquad (15)$$

$$\mathbf{s} = \text{softmax}\left(\sigma\left(\eta(\mathbf{h}_u^{(\ell)} \oplus \mathbf{h}_v^{(\ell)})\right)\right), \qquad (16)$$

where η is a non-linear neural network that computes a score vector $\mathbf{s} \in \mathbb{R}^{p_{\text{high}}}$ that gives the relative significance of each dimension for each message. This score vector is then used to select the top p dimensions for the message aggregation.

In our **PANDA**, it is important to emphasize that nodes with low and high-dimensional features are processed in their respective dimensional spaces throughout the network layers. This means that until the final layer, these nodes maintain their distinct dimensionalities.

Instance of our framework. To better understand our framework, we show how to integrate **PANDA** into two classical MPNNs: GCN (Kipf & Welling, 2017) and GIN (Xu et al., 2019). We will use these variations for our experiments. For the PANDA-GCN variant, the layer-update is expressed as follows for low and high-dimensional nodes. First, if the node v is low-dimensional:

$$\mathbf{h}_v^{(\ell+1)} = \mathbf{h}_v^{(\ell)} + \sigma\Big(\sum_{u \in \mathcal{N}_{\text{low}\leftrightarrow\text{low}}(v)} \frac{1}{\sqrt{d_u d_v}} \mathbf{W}_{\text{low}}^{(\ell)} \mathbf{h}_u^{(\ell)}$$
$$+ \sum_{u \in \mathcal{N}_{\text{low}\rightarrow\text{high}}(v)} \frac{1}{\sqrt{d_u d_v}} \mathbf{W}_{\text{low}}^{(\ell)} g(\mathbf{h}_v^{(\ell)}, \mathbf{h}_u^{(\ell)})\Big), \qquad (17)$$

then if node v is high-dimensional:

$$\mathbf{h}_v^{(\ell+1)} = \mathbf{h}_v^{(\ell)} + \sigma\Big(\sum_{u \in \mathcal{N}_{\text{high}\leftrightarrow\text{high}}(v)} \frac{1}{\sqrt{d_u d_v}} \mathbf{W}_{\text{high}}^{(\ell)} \mathbf{h}_u^{(\ell)}$$
$$+ \sum_{u \in \mathcal{N}_{\text{high}\rightarrow\text{low}}(v)} \frac{1}{\sqrt{d_u d_v}} \mathbf{W}_{\text{high}}^{(\ell)} f(\mathbf{h}_u^{(\ell)})\Big), \qquad (18)$$

where σ is a ReLU activation function, $\mathbf{W}_{\text{low}}^{(\ell)} \in \mathbb{R}^{p \times p}$ and $\mathbf{W}_{\text{high}}^{(\ell)} \in \mathbb{R}^{p_{\text{high}} \times p_{\text{high}}}$ are the weight matrices. The messages are normalized by their degrees, d_v and d_u. For the PANDA-GIN variant, we describe the layer-update in Appendix D.

Implementation. For implementation, the hidden vectors of different widths are separated and entered into the layer. These low and high-dimensional nodes are output with the same dimension in the last layer and integrated again.

3.3. Properties of PANDA

Directivity. Nodes with an equal width remain undirected, while edges among nodes with different widths can be directed. $\psi_{\twoheadrightarrow}^{(\ell)}$ and $\psi_{\rightarrowtail}^{(\ell)}$ have independent sets of parameters — Eqs. (17) and (18) have different weight parameters for low-dimensional and high-dimensional nodes. By ensuring the directivity only for edges with different widths, our method can be more expressive than its undirected counterpart (Rossi et al., 2023, Theorems 4.1 and 4.2).

Relational graph. Our **PANDA** can be considered as a relational graph convolutional network (R-GCN) (Schlichtkrull et al., 2018) applied to an augmented relational graph that incorporates two types of relation. Not only is message propagation performed according to the relationship of the neighbour set, but also different weight matrices are used for low-dim nodes and high-dim nodes, as mentioned earlier. In this sense, our **PANDA** is similar to R-GCN. However, R-GCN is not a model designed to alleviate over-squashing.

Computational complexity. Compared to existing rewiring methods, **PANDA** adds complexity from centrality calculation and four different message passings instead of graph rewiring. The specific complexity depends on which algorithm is used to calculate the centrality. For example, the time complexity of degree centrality is $\mathcal{O}(|\mathcal{E}|)$, and for betweenness centrality it is $\mathcal{O}(|\mathcal{V}|^3)$. We will discuss the empirical runtime of **PANDA** in Appendix H.

4. Discussions

This section discusses why our **PANDA** alleviates over-squashing from empirical perspectives, i.e., feature sensitivity, signal propagation, and Dirichlet energy.

Empirical sensitivity analysis. In order to verify that our method actually improves the feature sensitivity among nodes, we analyze the sensitivity given by Eq. (3) on benchmark datasets. As shown in Fig. 5, the sensitivity improves as the width p increases. Our method benefits from selectively having 64 and 128-dimensional nodes, yielding sensitivity that lies between these two dimensions — 16.7% of nodes have 128 dimensions in Fig. 5. Compared to other rewiring techniques, **PANDA** shows higher sensitivity that is maintained even in deeper layers, while others exhibit decreasing trends.
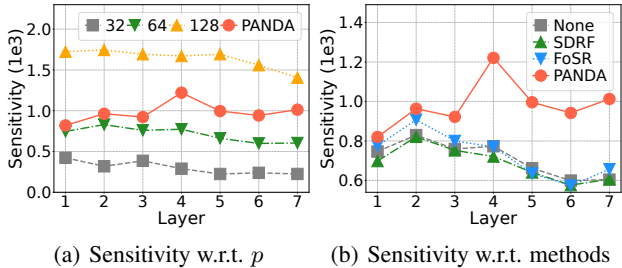
(a) Sensitivity w.r.t. p (b) Sensitivity w.r.t. methods

Figure 5. Empirical sensitivity across layers for GCN on MUTAG. More results on other datasets are in Appendix G.

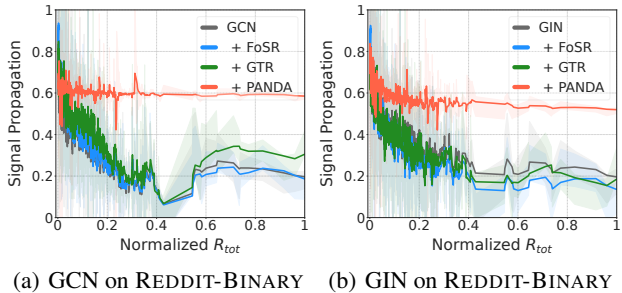(a) GCN on REDDIT-BINARY   (b) GIN on REDDIT-BINARY

Figure 6. The amount of signal propagated across the graph w.r.t. the normalized total effective resistance (R_{tot}). More results in other datasets are in Appendix G.

Signal propagation w.r.t. effective resistance. Di Giovanni et al. (2023) provide an empirical evidence that the information propagation of general GNNs is inversely proportional to the total effective resistance R_{tot}, which motivates us to check if our **PANDA** maintains the magnitude of signal flow in a graph with high R_{tot}. We further analyze if the signal flow improves after applying various rewiring methods. The detailed setting on the analysis is in Appendix A.2.

In Fig. 6, all methods except **PANDA** present decaying signal propagation trends as the resistance of graphs increases. In contrast, **PANDA** shows consistent signal propagation even for graphs with higher effective resistance. To verify the linear relationship between total effective resistance and signal propagation, we also quantify the correlation coefficient between them in Fig. 2. This result demonstrates the powerful effect of the width expansion in mitigating the over-squashing problem. Our **PANDA** maintains continuous information flows even under high bottleneck conditions, which cannot be overcome even by the direct modification of graph topology through various rewiring strategies.

Trade-off between rewiring and over-smoothing. As mentioned in Section 2.2, graph rewiring can cause over-smoothing. Since **PANDA** does not modify the original graph connectivity, we analyze whether our method
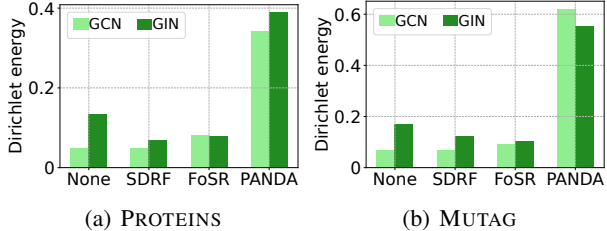

(a) PROTEINS   (b) MUTAG

Figure 7. Dirichlet energy of baselines and PANDA. More results in other datasets are in Appendix G.

can avoid the over-smoothing problem induced by graph rewiring using the Dirichlet energy as a measure of over-smoothing. The theoretical relation between Dirichelt energy and over-smoohting is given in Appendix E. Fig. 7 compares the Dirichlet energies of the final hidden representations learned from GCN and GIN having both a rewiring technique and **PANDA**. The result shows that **PANDA** yields a higher Dirichlet energy for both GCN and GIN compared to others, highlighting the strength of our model, mitigating the over-squashing while preserving the original graph connectivity.

5. Experiments

In this section, we empirically verify that **PANDA** can significantly improve the performance of GNN over other rewiring methods. We experiment with graph classification and node classification, as well as tasks on Long-Range Graph Benchmark (LRGB) (Dwivedi et al., 2022). We cover the experiments on TUDataset (Morris et al., 2020) in the main content and the node classification task and LRGB dataset in Appendix K and Appendix L.

Datasets. For graph classification, we report our method using the following benchmarks: REDDIT-BINARY, IMDB-BINARY, MUTAG, ENZYMES, PROTEINS, COLLAB from the TUDataset (Morris et al., 2020). A detail of datasets is available in Appendix F.1.

Experimental setting. For graph classification, we compare **PANDA** to no graph rewiring and 7 other state-of-the-art rewiring methods: DIGL (Gasteiger et al., 2019), Fully adjacent layer (FA) (Alon & Yahav, 2021), SDRF (Topping et al., 2022), FoSR (Karhadkar et al., 2023), BORF (Karhadkar et al., 2023), GTR (Black et al., 2023), and CT-Layer (Arnaiz-Rodríguez et al., 2022). In our experiments, we prioritize fairness and comprehensiveness rather than aiming to obtain the best possible performance for each dataset. For backbone GNNs, we use GCN, GIN (Xu et al., 2019), R-GCN (Schlichtkrull et al., 2018), and R-GIN (Brockschmidt, 2020). We accumulate the result in 100 random trials and report the mean test accuracy, along

Table 1. Results of PANDA and baselines for GCN and GIN. We show the best three in red (first), blue (second), and purple (third).

Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
GCN (None)	68.255 ± 1.098	49.770 ± 0.817	72.150 ± 2.442	27.667 ± 1.164	70.982 ± 0.737	33.784 ± 0.488
+ Last Layer FA	68.485 ± 0.945	48.980 ± 0.945	70.050 ± 2.027	26.467 ± 1.204	71.018 ± 0.963	33.320 ± 0.435
+ Every Layer FA	48.490 ± 1.044	48.170 ± 0.801	70.450 ± 1.960	18.333 ± 1.038	60.036 ± 0.925	51.798 ± 0.419
+ DIGL	49.980 ± 0.680	49.910 ± 0.841	71.350 ± 2.391	27.517 ± 1.053	70.607 ± 0.731	15.530 ± 0.294
+ SDRF	68.620 ± 0.851	49.400 ± 0.904	71.050 ± 1.872	28.367 ± 1.174	70.920 ± 0.792	33.448 ± 0.472
+ FoSR	70.330 ± 0.727	49.660 ± 0.864	80.000 ± 1.574	25.067 ± 0.994	73.420 ± 0.811	33.836 ± 0.584
+ BORF	Time-out	50.100 ± 0.900	75.800 ± 1.900	24.700 ± 1.000	71.000 ± 0.800	Time-out
+ GTR	68.990 ± 0.610	49.920 ± 0.990	79.100 ± 1.860	27.520 ± 0.990	72.590 ± 2.480	33.050 ± 0.400
+ CT-Layer	51.580 ± 1.019	50.320 ± 0.944	75.899 ± 3.024	17.383 ± 1.030	60.357 ± 1.060	52.146 ± 0.415
+ **PANDA**	80.690 ± 0.721	63.760 ± 1.012	85.750 ± 1.396	31.550 ± 1.230	76.000 ± 0.774	68.400 ± 0.452
GIN (None)	86.785 ± 1.056	70.180 ± 0.992	77.700 ± 0.360	33.800 ± 0.115	70.804 ± 0.827	72.992 ± 0.384
+ Last Layer FA	90.220 ± 0.475	70.910 ± 0.788	83.450 ± 1.742	47.400 ± 1.387	72.304 ± 0.666	75.056 ± 0.406
+ Every Layer FA	50.360 ± 0.684	49.160 ± 0.870	72.550 ± 3.016	28.383 ± 1.052	70.375 ± 0.910	32.984 ± 0.390
+ DIGL	76.035 ± 0.774	64.390 ± 0.907	79.700 ± 2.150	35.717 ± 1.198	70.759 ± 0.774	54.504 ± 0.410
+ SDRF	86.440 ± 0.590	69.720 ± 1.152	78.400 ± 2.803	35.817 ± 1.094	69.813 ± 0.792	72.958 ± 0.419
+ FoSR	87.350 ± 0.598	71.210 ± 0.919	78.400 ± 2.803	29.200 ± 1.367	75.107 ± 0.817	73.278 ± 0.416
+ BORF	Time-out	71.300 ± 1.500	80.800 ± 2.500	35.500 ± 1.200	74.200 ± 0.800	Time-out
+ GTR	86.980 ± 0.660	71.280 ± 0.860	77.600 ± 2.840	30.570 ± 1.420	73.130 ± 0.690	72.930 ± 0.420
+ CT-Layer	54.589 ± 1.757	50.000 ± 0.974	56.850 ± 4.253	16.583 ± 0.907	61.107 ± 1.184	52.304 ± 0.605
+ **PANDA**	91.055 ± 0.402	72.560 ± 0.917	88.750 ± 1.570	46.200 ± 1.410	75.759 ± 0.856	75.110 ± 0.210

Table 2. PANDA-GCN v.s. R-GCN

Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
R-GCN	49.850 ± 0.653	50.012 ± 0.917	69.250 ± 2.085	28.600 ± 1.186	69.518 ± 0.725	33.602 ± 1.047
PANDA-GCN	$\mathbf{80.690 \pm 0.721}$	$\mathbf{63.760 \pm 1.012}$	$\mathbf{85.750 \pm 1.396}$	$\mathbf{31.550 \pm 1.230}$	$\mathbf{76.000 \pm 0.774}$	$\mathbf{68.400 \pm 0.452}$

with the 95% confidence interval. Further experiment details are in Appendix F.

Results. Table 1 shows the results of different methods applied to the GCN and GIN models across benchmark datasets. Our **PANDA** method shows the highest accuracy across most of the datasets for both models, significantly outperforming the baseline ("None") and other rewiring methods. Surprisingly, For REDDIT-BINARY, our **PANDA** shows a substantial 22.30% improvement over the baseline (None) method. In particular, in the case of BORF, it is impracticable to evaluate for REDDIT-BINARY and COLLAB due to the rewiring time. In the case of GIN, our **PANDA** also leads to the highest accuracies, with 91.055 on REDDIT-BINARY and 88.75 on MUTAG. These results underscore the effectiveness of our **PANDA** in enhancing the performance of GNN. In Table 9 of Appendix I.1, we also show the experimental results for R-GCN and R-GIN.

Performance comparison of PANDA-GCN and R-GCN.
Considering the similarities between PANDA-GCN and R-GCN discussed in the previous section, we empirically evaluate their performance. As shown in Table 2, PANDA-GCN outperforms R-GCN in all datasets. Compared to R-GCN, which is not designed to mitigate over-squashing,

these results show the efficacy of **PANDA**.

Effect of centrality metrics. It is natural to ask how different the performance of **PANDA** varies depending on the centrality metrics. Table 3 compares the results obtained using different kinds of centrality metrics. In IMDB-BINARY, PageRank centrality is better than other metrics, indicating that larger node widths are effective in receiving messages from influential nodes. Closeness centrality shows the best performance on PROTEINS and COLLAB. This indicates that it is effective to expand the width of nodes that require many connections to connect to other distant nodes.

Sensitivity on top-k nodes. Fig. 8 shows a sensitivity study w.r.t. top-k nodes. For PROTEINS, both PANDA-GCN and PANDA-GIN show an increase in mean accuracy as the top-k value increased from 3 to 7. However, the accuracy does not increase after the k value is 7. For MUTAG, PANDA-GCN and PANDA-GIN have the highest accuracy for k values of 10 and 7, respectively.

Sensitivity on p_{high}. Fig. 9 shows a sensitivity study of the **PANDA** w.r.t. p_{high}. In PROTEINS, using PANDA-GCN and PANDA-GIN with larger p_{high} improves performance. In MUTAG, on the other hand, both models achieve their

Table 3. Performance comparison by various centrality measures for PANDA-GCN. The results of PANDA-GIN are in Appendix I.2.

$C(\mathcal{G})$	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
Degree	80.690 ± 0.721	62.100 ± 1.043	85.200 ± 1.568	31.117 ± 1.258	75.375 ± 0.800	68.162 ± 0.471
Betweenness	80.000 ± 0.659	59.630 ± 1.152	85.750 ± 1.396	29.600 ± 1.208	74.589 ± 0.791	67.844 ± 0.547
Closeness	79.700 ± 0.664	61.160 ± 0.992	84.700 ± 1.554	29.967 ± 1.231	76.000 ± 0.774	68.400 ± 0.452
PageRank	80.340 ± 0.826	63.760 ± 1.012	85.450 ± 1.569	31.550 ± 1.230	74.098 ± 0.851	67.540 ± 0.500
Load	79.500 ± 0.732	59.840 ± 1.153	85.700 ± 1.549	28.167 ± 1.090	74.188 ± 0.814	67.802 ± 0.506

(a) PROTEINS   (b) MUTAG
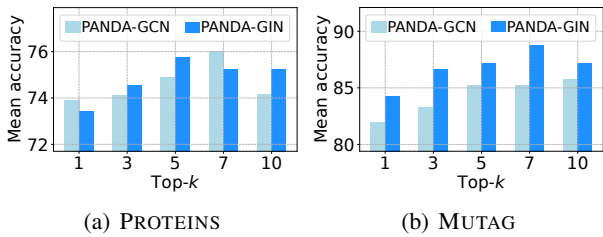
Figure 8. Sensitivity on top-k. More results are in Appendix J.

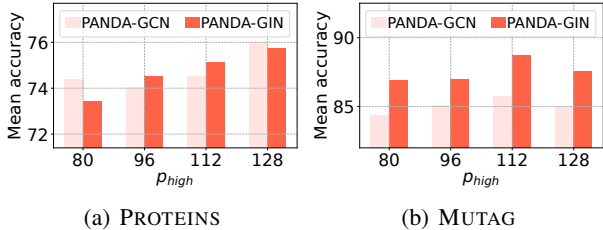(a) PROTEINS   (b) MUTAG

Figure 9. Sensitivity on p_{high}. More results are in Appendix J.

best performance when using a p_{high} value of 112.

6. Related Work

This section introduces related work to alleviate the over-squashing problem. Before the link between over-squashing and graph curvature was established by (Topping et al., 2022), DIGL was developed to sparsify the adjacency matrix using the graph heat kernel and personalized PageRank (Gasteiger et al., 2019). After Alon & Yahav (2021) empirically observed the over-squashing problem, research is active to find indicators for over-squashing and propose rewiring methods. Topping et al. (2022) initially connect over-squashing with graph Ricci curvature, showing that edges with highly negative Ricci curvature contribute to the over-squashing and proposes an SDRF. Since then, rewiring methods with indicators based on curvatures have been studied (Nguyen et al., 2023; Shi et al., 2023a). In addition to using curvatures as indicators of the over-squashing, Black et al. (2023) measure the over-squashing through the concept of effective resistance. Banerjee et al. (2022) propose rewiring methods to handle the original topology of the graph to prevent it from being disconnected (Banerjee et al.,

2022). To prevent over-smoothing, Karhadkar et al. (2023) propose a rewiring method that optimizes the spectral gap. Deac et al. (2022) propose to interleave message propagation on the original graph with message passing on an expander graph to alleviate over-squashing. Gutteridge et al. (2023) propose a layer-dependent rewiring method that provides a variety of rewiring graphs for feature propagation. Barbero et al. (2023) propose a LASER to rewire the graph to better preserve locality. Finkelshtein et al. (2023) propose a CO-GNN with flexible and dynamic message passing that can perform effective graph rewiring at each layer of GNN. Recently, Di Giovanni et al. (2023) analyze the factors that contribute to over-squashing, and Southern et al. (2024) analyze the role of virtual nodes in over-squashing. Also, rewiring methods, as well as more advanced and flexible message-passing paradigms, are being studied (Errica et al., 2023; Barbero et al., 2023; Park et al., 2023; Qian et al., 2024; Behrouz & Hashemi, 2024; Qiu et al., 2024).

Most studies to mitigate over-squashing focus on rewiring methods that change the topology. However, PANDA is different in that it addresses the over-squashing problem by moving away from the rewiring method.

7. Concluding Remarks

We presented **PANDA**, a novel message passing framework, effectively addressing the over-squashing problem in GNNs without rewiring edges. By selectively expanding the width of high-centrality nodes, **PANDA** promotes signal propagation to alleviate over-squashing. Our empirical evaluations show that **PANDA** outperforms existing graph rewiring methods in graph classification. This research contributes a pioneering approach to selectively expanding node widths in message passing and lays the groundwork for the future exploration of width-aware strategies in GNNs.

Limitations and future work. We only focus on redesigning the GNN based on sensitivity (Eq. (3)) for the width of the nodes that contribute to over-squashing. Despite our empirical evidence for mitigating over-squashing, more research is needed to prove the effects of higher widths on over-squashing. For future work, we will aim to reduce the complexity of our method and explore a much wider range of tasks to study the pros and cons of higher widths.

Impact Statement

Our proposed **PANDA** is designed as a general graph representation learning method. Therefore, our proposed method has no discernible negative impact on society. Nonetheless, adverse or malicious applications of the proposed algorithms in various domains, including drug discovery and healthcare, may lead to undesirable effects.

Acknowledgements

References

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=i80OPhOCVH2.

Arnaiz-Rodríguez, A., Begga, A., Escolano, F., and Oliver, N. M. Diffwire: Inductive graph rewiring via the lovász bound. In *Learning on Graphs Conference*, 2022. URL https://proceedings.mlr.press/v198/arnaiz-rodri-guez22a.html.

Banerjee, P. K., Karhadkar, K., Wang, Y. G., Alon, U., and Montúfar, G. Oversquashing in gnns through the lens of information contraction and graph expansion. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1–8. IEEE, 2022.

Barbero, F., Velingker, A., Saberi, A., Bronstein, M., and Di Giovanni, F. Locality-aware graph-rewiring in gnns. *arXiv preprint arXiv:2310.01668*, 2023.

Behrouz, A. and Hashemi, F. Graph mamba: Towards learning on graphs with state space models. *arXiv preprint arXiv:2402.08678*, 2024.

Black, M., Wan, Z., Nayyeri, A., and Wang, Y. Understanding oversquashing in GNNs through the lens of effective resistance. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/black23a.html.

Borgatti, S. P. and Halgin, D. S. Analyzing affiliation networks. *The Sage handbook of social network analysis*, 1: 417–433, 2011.

Brockschmidt, M. Gnn-film: Graph neural networks with feature-wise linear modulation. In *International Conference on Machine Learning*, pp. 1144–1152. PMLR, 2020.

Chamberlain, B. P., Rowbottom, J., Goronova, M., Webb, S., Rossi, E., and Bronstein, M. M. GRAND: Graph neural diffusion. In *International Conference on Machine Learning*, 2021.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, 2020.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*, 2021.

Choi, J. and Park, N. Graph neural rough differential equations for traffic forecasting. *ACM Transactions on Intelligent Systems and Technology*, 14(4):1–27, 2023.

Choi, J., Jeon, J., and Park, N. LT-OCF: Learnable-time ode-based collaborative filtering. In *CIKM*, 2021.

Choi, J., Hong, S., Park, N., and Cho, S.-B. Gread: Graph neural reaction-diffusion networks. In *International Conference on Machine Learning*, pp. 5722–5747, 2023.

Deac, A., Lackenby, M., and Veličković, P. Expander graph propagation. In *Learning on Graphs Conference*, pp. 38–1. PMLR, 2022.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 2016.

Devriendt, K. and Lambiotte, R. Discrete curvature on graphs from the effective resistance. *Journal of Physics: Complexity*, 3(2):025008, 2022.

Di Giovanni, F., Giusti, L., Barbero, F., Luise, G., Lio, P., and Bronstein, M. M. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/di-giovanni23a.html.

Dwivedi, V. P., Rampášek, L., Galkin, M., Parviz, A., Wolf, G., Luu, A. T., and Beaini, D. Long range graph benchmark. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=in7XC5RcjEn.

Errica, F., Christiansen, H., Zaverkin, V., Maruyama, T., Niepert, M., and Alesiani, F. Adaptive message passing: A general framework to mitigate oversmoothing, oversquashing, and underreaching. *arXiv preprint arXiv:2312.16560*, 2023.

Fan R. K. Chung, F. C. G. *Spectral graph theory*. Number 92. 1997.

Fesser, L. and Weber, M. Mitigating over-smoothing and over-squashing using augmentations of forman-ricci curvature. *arXiv preprint arXiv:2309.09384*, 2023.

Finkelshtein, B., Huang, X., Bronstein, M., and Ceylan, İ. İ. Cooperative graph neural networks. *arXiv preprint arXiv:2310.01267*, 2023.

Freeman, L. C. A set of measures of centrality based on betweenness. *Sociometry*, pp. 35–41, 1977.

Freeman, L. C. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978.

Gasteiger, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. In *Advances in neural information processing systems*, 2019.

Ghosh, A., Boyd, S., and Saberi, A. Minimizing effective resistance of a graph. *SIAM review*, 50(1):37–66, 2008.

Giraldo, J. H., Skianis, K., Bouwmans, T., and Malliaros, F. D. 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*, pp. 566–576, 2023.

Goh, K.-I., Kahng, B., and Kim, D. Universal behavior of load distribution in scale-free networks. *Physical review letters*, 87(27):278701, 2001.

Gruber, A., Lee, K., and Trask, N. Reversible and irreversible bracket-based dynamics for deep graph neural networks. *Advances in Neural Information Processing Systems*, 36, 2023.

Gutteridge, B., Dong, X., Bronstein, M. M., and Di Giovanni, F. Drew: Dynamically rewired message passing with delay. In *International Conference on Machine Learning*, pp. 12252–12267. PMLR, 2023.

Hwang, J., Choi, J., Choi, H., Lee, K., Lee, D., and Park, N. Climate modeling with neural diffusion equations. In *ICDM*, pp. 230–239, 2021.

Kang, Q., Zhao, K., Song, Y., Wang, S., and Tay, W. P. Node embedding from neural hamiltonian orbits in graph neural networks. In *International Conference on Machine Learning*, pp. 15786–15808. PMLR, 2023.

Karhadkar, K., Banerjee, P. K., and Montufar, G. FoSR: First-order spectral rewiring for addressing oversquashing in GNNs. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3YjQfCLdrzz.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Li, Q., Han, Z., and Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020. URL www.graphlearning.io.

Nguyen, K., Hieu, N. M., Nguyen, V. D., Ho, N., Osher, S., and Nguyen, T. M. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature. In *International Conference on Machine Learning*, 2023. URL https://proceedings.mlr.press/v202/nguyen23c.html.

Nt, H. and Maehara, T. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Oono, K. and Suzuki, T. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

Park, S., Ryu, N., Kim, G., Woo, D., Yun, S.-Y., and Ahn, S. Non-backtracking graph neural networks. *arXiv preprint arXiv:2310.07430*, 2023.

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y., and Yang, B. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*, 2020.

Qian, C., Manolache, A., Ahmed, K., Zeng, Z., den Broeck, G. V., Niepert, M., and Morris, C. Probabilistically rewired message-passing neural networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Tj6Wcx7gVk.

Qiu, H., Bian, Y., and Yao, Q. Graph unitary message passing. *arXiv preprint arXiv:2403.11199*, 2024.

Rossi, E., Charpentier, B., Di Giovanni, F., Frasca, F., Günnemann, S., and Bronstein, M. Edge directionality improves learning on heterophilic graphs. *arXiv preprint arXiv:2305.10498*, 2023.

Rozemberczki, B., Allen, C., and Sarkar, R. Multi-Scale Attributed Node Embedding. *Journal of Complex Networks*, 9(2), 2021.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pp. 593–607. Springer, 2018.

Shi, D., Guo, Y., Shao, Z., and Gao, J. How curvature enhance the adaptation power of framelet gcns. *arXiv preprint arXiv:2307.09768*, 2023a.

Shi, D., Han, A., Lin, L., Guo, Y., and Gao, J. Exposition on over-squashing problem on GNNs: Current methods, benchmarks and challenges. *arXiv preprint arXiv:2311.07073*, 2023b.

Sonthalia, R., Gilbert, A., and Durham, M. RelWire: Metric based graph rewiring. In *Advances in Neural Information Processing Systems 2023 Workshop on Symmetry and Geometry in Neural Representations*, 2023. URL https://openreview.net/forum?id=c9u8tH1WA0.

Southern, J., Di Giovanni, F., Bronstein, M., and Lutzeyer, J. F. Understanding virtual nodes: Oversmoothing, oversquashing, and node heterogeneity. *arXiv preprint arXiv:2405.13526*, 2024.

Thomassen, C. Resistances and currents in infinite electrical networks. *Journal of Combinatorial Theory, Series B*, 49 (1):87–102, 1990.

Topping, J., Giovanni, F. D., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=7UmjRGzp-A.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph Attention Networks. In *International Conference on Learning Representations*, 2018.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Yang, Z., Cohen, W. W., and Salakhutdinov, R. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*, 2016.

Yu, H., Kim, P. M., Sprecher, E., Trifonov, V., and Gerstein, M. The importance of bottlenecks in protein networks: correlation with gene essentiality and expression dynamics. *PLoS computational biology*, 3(4):e59, 2007.

Supplementary Materials for "PANDA: Expanded Width-Aware Message Passing Beyond Rewiring"

A. Effective Resistance and Signal Propagation

A.1. Effective Resistance

Total effective resistance. The resistance between the nodes u and v in the graph is given by

$$R_{u,v} = (\mathbf{1}_u - \mathbf{1}_v)^\mathsf{T} \mathbf{L}^+ (\mathbf{1}_u - \mathbf{1}_v), \tag{19}$$

where $\mathbf{1}_v$ and $\mathbf{1}_u$ are indicator vectors for node u and v, respectively. Total effective resistance, R_{tot}, is defined as the sum of effective resistance between all pairs of nodes (Ghosh et al., 2008; Black et al., 2023):

$$R_{tot} = \sum_{u > v} R_{u,v} = n \cdot \mathrm{Tr}(\mathbf{L}^+) = n \sum_i^n \frac{1}{\lambda_i}, \tag{20}$$

where λ_i is the i-th eigenvalues of $\mathbf{L}$ and $\mathbf{L}^+$ is the pseudoinverse of $\mathbf{L}$.

A.2. Signal Propagation

Signal propagation w.r.t. effective resistance. Here, we provide experimental details for measuring signal propagation w.r.t. normalized total effective resistance of the graphs. First, fix a source node $v \in \mathcal{V}$ and assign p-dimensional feature vector to it, while assign the rest of the nodes zero vectors. Then, the amount of signal that has been propagated over the graph by the randomly initialized model with ℓ layers is given by

$$\mathbf{h}_\odot^{(\ell)} = \frac{1}{p \max_{u \neq v} k_\mathcal{G}(u,v)} \sum_{t=1}^p \sum_{u \neq v} \frac{\mathbf{h}_u^{(\ell),t}}{\|\mathbf{h}_u^{(\ell),t}\|} k_\mathcal{G}(u,v), \tag{21}$$

where $\mathbf{h}_u^{(\ell),t}$ is the t-th feature of p-dimensional feature vector of node u at layer ℓ and $k_\mathcal{G}(u,v)$ is the distance between two nodes u and v, computed as a shortest path. Every unitary signal $\mathbf{h}_u^{(\ell),t}/\|\mathbf{h}_u^{(\ell),t}\|$ propagated across the graph G from the source node v is weighted by the normalized propagation distance $k_\mathcal{G}(u,v)/\max_{u \neq v} d_G(u,v)$ for all nodes $u \neq v$ and then averaged over entire p output channels. 10 nodes are randomly sampled from each graph and total effective resistance of the graph is estimated for each source node. The final $\mathbf{h}_\odot^{(\ell)}$ and total resistance of the graph are calculated as the mean of all 10 nodes. The experiment is repeated for every graph in the dataset and the signal propagation measured for each graph is plotted w.r.t. the normalized total effective resistance of the corresponding graph.

B. Proposition on Enhanced Sensitivity to Alleviate Over-Squashing

We provide a proposition based on Eq. (3) that the introduction can alleviate the over-squashing.

Proposition B.1 (Informal). *Given the sensitivity bound in Eq. (3), we consider two scenarios: one with a standard width p and the other with an increased width p_{high}, where $p_{\mathrm{high}} > p$. For nodes with increased width, the sensitivity bound is augmented, demonstrating a potentially higher sensitivity to input features:*

$$\left\| \frac{\partial \mathbf{h}_v^{(\ell)}}{\partial \mathbf{h}_u^{(0)}} \right\|_1 \leq (zwp)^\ell \left(\mathbf{S}^\ell \right)_{vu} < (zwp_{\mathrm{high}})^\ell \left(\mathbf{S}^\ell \right)_{vu}. \tag{22}$$

This increased sensitivity can potentially reduce over-squashing, facilitating better signal propagation.

Proposition B.1 provides the theoretical foundation for our approach. We are inspired by the insights gained from increased upper bound with increased width, but we emphasize that we do not increase the width of all nodes, but rather *selectively*.

C. Graph Centrality Metrics

We next introduce five graph centrality metrics:

- Degree centrality (Borgatti & Halgin, 2011) measures the number of direct connections a node has to other nodes in the network, e.g., the degree of nodes. The centrality value of node v is the fraction of nodes connected to it. This centrality measures a local characteristic and does not take into account global topology or neighbor connectivity.

- PageRank centrality (Page et al., 1999) measures the importance of a node based on the importance of its neighbors. It is an iterative algorithm that assigns a node a high score if it is connected to many nodes that themselves have high scores. High PageRank centrality indicates that a node is not only connected to many nodes but also to nodes that are themselves central or important within the network. This centrality uses a damping factor to control neighbors' effect on your node while measuring its importance. We set the damping factor as 0.85.

- Closeness centrality (Freeman, 1978) is a metric that measures how close a particular node is to all other nodes within a network. This metric is calculated based on the shortest path length between nodes in a network and indicates how efficiently a node is connected to other nodes in the network on average. Closeness centrality of a node v is the reciprocal of the average shortest path distance to u over all $n-1$ reachable nodes. Nodes with high closeness centrality are located relatively close to all other nodes in the network. Because a node with high closeness centrality plays a central role through the shortest paths to many other nodes, the amount of information focused on this node can be very large.

- Betweenness centrality (Freeman, 1977) measures the extent to which a node lies on the shortest paths between other nodes. It highlights nodes that serve as bridges or points of control within the network. A node with high betweenness centrality significantly influences the flow of information or resources within the network, as it can affect the transfer between other nodes by facilitating or constraining it. For this reason, in studies that aim to measure over-squashing (Topping et al., 2022), it is also used to measure the bottleneck of the graph. The centrality score of a node is calculated by the number of shortest paths that include the node. However, this metric is notorious for its high complexity to calculate all-pair shortest paths.

- Load centrality (Goh et al., 2001) is similar to betweenness centrality but considers the number of times a node is traversed by all shortest paths in the network. It is a measure of the load or traffic that a node will handle. Nodes with high load centrality are critical for the flow of information or resources, as they are likely to be bottlenecks or critical points of failure within the network. Like other centralities, it is useful for finding nodes where bottlenecks occur.

D. Layer-update of PANDA-GIN

In this section, we define the layer-update of PANDA-GIN. If the node v is low-dimensional:

$$\mathbf{h}_v^{(\ell+1)} = \text{MLP}_{\text{low}}^{(\ell)}\Big((1+\epsilon)\mathbf{h}_v^{(\ell)} + \sum_{u\in\mathcal{N}_{\text{low}\leftrightarrow\text{low}}(v)} \mathbf{h}_u^{(\ell)} + \sum_{u\in\mathcal{N}_{\text{high}\rightarrowtail\text{low}}(v)} g(\mathbf{h}_v^{(\ell)}, \mathbf{h}_u^{(\ell)})\Big), \tag{23}$$

where $\text{MLP}_{\text{low}}^{(\ell)}$ is one or more linear layers separated by ReLU activation and ϵ is a weight parameter. For the high-dimensional nodes, the layer-update is defined as follows:

$$\mathbf{h}_v^{(\ell+1)} = \text{MLP}_{\text{high}}^{(\ell)}\Big((1+\epsilon)\mathbf{h}_v^{(\ell)} + \sum_{u\in\mathcal{N}_{\text{high}\leftrightarrow\text{high}}(v)} \mathbf{h}_u^{(\ell)} + \sum_{u\in\mathcal{N}_{\text{low}\rightarrowtail\text{high}}(v)} f(\mathbf{h}_u^{(\ell)})\Big). \tag{24}$$

E. Dirichlet Energy and Over-smoothing

Let $\mathcal{G}$ be a graph with adjacency matrix $\mathbf{A}$ and normalized Laplacian $\widetilde{\mathbf{L}} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. Given a vector field $\mathbf{H} \in \mathbb{R}^{n\times p}$ defined on nodes in $\mathcal{G}$, Dirichlet energy is defined as

$$E_{\text{Dir}}(\mathbf{H}) := \text{Tr}(\mathbf{H}^\mathsf{T}\widetilde{\mathbf{L}}\mathbf{H}) = \frac{1}{2}\sum_{v,u,w} \mathbf{A}_{v,u}\left(\frac{\mathbf{H}_{v,u}}{\sqrt{d_v}} - \frac{\mathbf{H}_{u,w}}{\sqrt{d_u}}\right)^2 \tag{25}$$

Essentially, the Dirichlet energy quantifies how much a function on the graph deviates from being constant between connected node pairs, which thereby indicates how non-smooth the signals on a graph are (Fan R. K. Chung, 1997; Arnaiz-Rodríguez et al., 2022). Hence a metric of Dirichlet energy has been widely applied to measure the amount of over-smoothing of the graph representations.

F. Experimental Details for Graph Classification

F.1. Dataset Statisctics

General statistics for graph classification datasets. We consider the REDDIT-BINARY (2,000 graphs), IMDB-BINARY (1,000 graphs), MUTAG (188 graphs), ENZYMES (600 graphs), PROTEINS (1,113 graphs), and COLLAB (5,000 graphs) tasks from TUDatasets (Morris et al., 2020). These datasets were chosen by Karhadkar et al. (2023), under the claim that they require long-range interactions.

F.2. Detail of Baselines

We compare our method with the following baselines and try to focus on baselines that preprocess the input graph by adding edges.

DIGL. Diffusion Improves Graph Learning (DIGL) (Gasteiger et al., 2019) is a diffusion-based rewiring scheme that computes a kernel evaluation of the adjacency matrix, followed by sparsification.

SDRF. Stochastic Discrete Ricci Flow (SDRF) (Topping et al., 2022) surgically rewires a graph by adding edges to support other edges with low curvature, which are locations of bottlenecks. For SDRF, we include results for both the original method (configured to only add edges), and our relational method (again only add edges and include the added edges with their own relation).

FA. Fully adjacent (FA) layers (Alon & Yahav, 2021) rewire the graph by adding all possible edges. We include results for rewiring only the last layer (last layer FA) and rewiring every layer (every layer FA).

FOSR. First-order Spectral Rewiring (FoSR) (Karhadkar et al., 2023) is a graph rewiring method for preventing over-squashing based on iterative first-order maximization of the spectral gap.

BORF. Batch Ollivier-Ricci Flow (BORF) (Karhadkar et al., 2023) is a novel curvature-based rewiring method designed to mitigate the over-smoothing and over-squashing issues simultaneously.

GTR. Greedy Total Resistance (GTR) (Black et al., 2023) is a rewiring method based on total effective resistance.

CT-Layer. Commute Time (CT)-Layer (Arnaiz-Rodríguez et al., 2022) is a method that learns the commute time and rewires the input graph accordingly. In case of CT-Layer, we isolated the CT-Layer that consists of CT pooling and CT rewiring steps from the DiffWire (Arnaiz-Rodríguez et al., 2022) where the input graph is rewired and re-weighted based on the learnable commute time embeddings. Then we subsequently add series of GCN and GIN layers identical to the ones used in other rewiring methods and **PANDA**.

F.3. Evaluation protocol

In order to faithfully compare the performance of the rewiring techniques, we follow the same setting as in Karhadkar et al. (2023). Each configuration is evaluated using the validation set. The testset accuracy of the configuration with the best validation performance is then recorded. For each experiment, we accumulate the result in 100 random trials with a 80%/10%/10% train/val/test split and report the mean test accuracy, along with the 95% confidence interval.

F.4. Hyperparameters

For each task and baseline model, we used the same settings of GNN and optimization hyperparameters across all methods to rule out hyperparameter tuning as a source of performance gain. Table 4 shows common hyperparameters. Table 5 shows the search range for hyperparameters of **PANDA**, and Table 6 shows the best hyperparameters used by **PANDA**.

F.5. Hardware specifications and libraries

The following software and hardware environments were used for all experiments: UBUNTU 18.04 LTS, PYTHON 3.7.13, PYTORCH 1.11.0, PYTORCH GEOMETRIC 2.0.4, NUMPY 1.21.6, NETWORKX 2.6.3, CUDA 11.3, and NVIDIA Driver

Table 4. Common hyperparameters

Common Hyperparameters	
Dropoout	0.5
Number of layers	4
Hidden dimension	64
Learning rate	0.001
Stopping patience	100 epochs

Table 5. Hyperparameter search space of PANDA for benchmark datasets

Hyperparameters	Search Space
p_{high}	{80, 96, 112, 128}
top k	{1, 3 ,5, 7, 10, 15, 20}
Centrality	{Degree, Betweenness, Closeness, PageRank, Load }

Table 6. Best hyperparameter

Hyperparameter	Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
	GCN	7	3	10	7	7	3
	GIN	10	3	10	10	5	10
Top-k	R-GCN	5	5	5	10	10	3
	R-GIN	15	5	5	15	5	10
	GCN	80	80	112	128	128	128
	GIN	96	128	96	128	112	96
p_{high}	R-GCN	128	128	128	96	112	128
	R-GIN	96	128	128	112	128	128
	GCN	Degree	PageRank	Betweenness	PageRank	Closeness	Degree
	GIN	PageRank	PageRank	PageRank	Closeness	Degree	Load
$C(\mathcal{G})$	R-GCN	Degree	PageRank	PageRank	Betweenness	Betweenness	Degree
	R-GIN	Degree	Degree	PageRank	Betweenness	Betweenness	Betweenness

465.19, and i9 CPU, and NVIDIA RTX 3090. We implemented our **PANDA** message passing framework in PYTORCH GEOMETRIC.

G. Additional Visualizations in Section 4

(a) GCN on ENZYMES   (b) GIN on ENZYMES   (c) GCN on MUTAG   (d) GIN on MUTAG

(e) GCN on PROTEINS   (f) GIN on PROTEINS   (g) GCN on REDDIT-BINARY   (h) GIN on REDDIT-BINARY
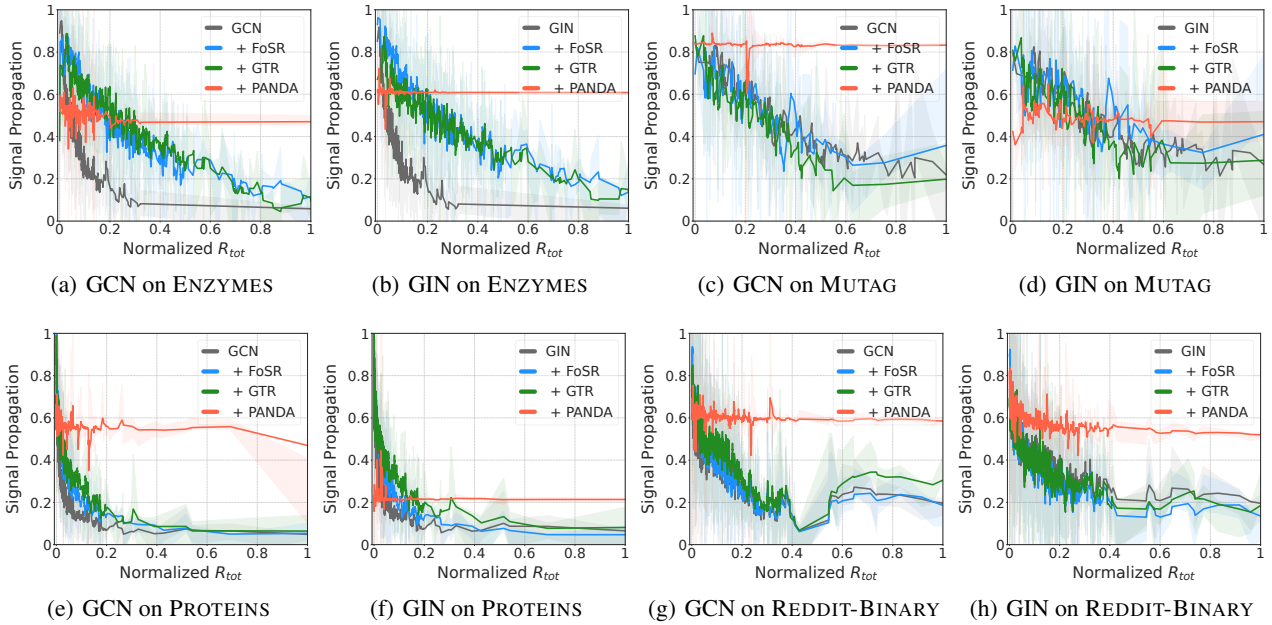
Figure 10. The amount of signal propagated across the graph w.r.t. the normalized total effective resistance (R_{tot}) for all datasets.
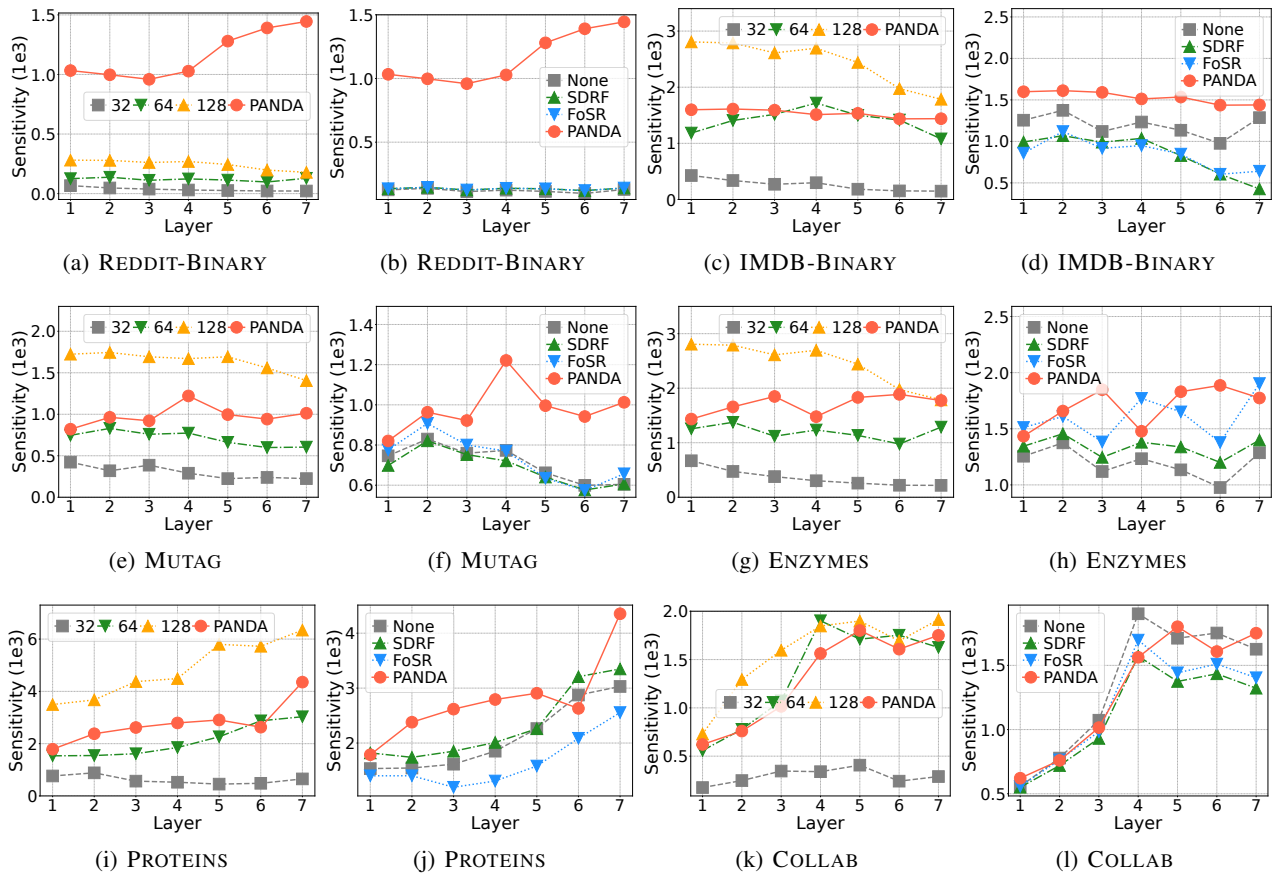
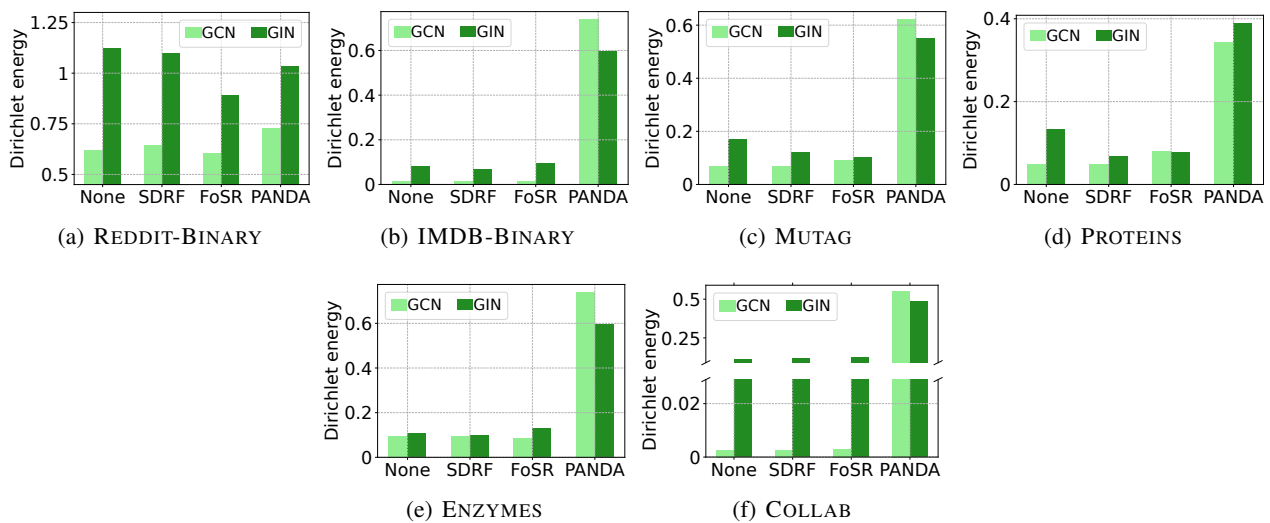Figure 11. Empirical sensitivity across layers for GCN on all datasets.

Figure 12. Dirichlet energy of baselines and PANDA.

H. Empirical Computational Complexity

H.1. Empirical Runtime Analysis

We compare the runtime against the centrality metric used by PANDA and the rewiring method FoSR. As can be seen in Table 7, using centrality metrics for IMDB-BINARY, MUTAG, and PROTEINS is faster than the rewiring method FoSR. However, when the size of the graph such as REDDIT-BINARY or COLLAB is large, metrics such as betweenness, closeness, and load centrality that calculate the shortest path take a long time.

In Table 8, we also measure and compare the runtime of the forward pass of the original GCN and PANDA-GCN. **PANDA** takes more time than the original GCN due to the amount of computation used in Eqs. (14) and (15). However, in our design, it is meaningful to select the different dimensions of high-dimensional nodes that are learned to be more significant for each message (edge) in $g(\cdot)$ of Eq. (15). In future work, we aim to design an efficient method for selecting the dimensions.

Table 7. Empirical runtime comparison of rewiring methods and centrality metrics (in seconds)

	Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
	FoSR	11.97	3.36	3.43	3.28	3.47	6.55
PANDA	Degree	3.33	0.24	0.03	0.14	0.27	21.05
	Betweenness	965.31	1.16	0.06	1.20	5.58	349.68
	Closeness	238.32	0.38	0.43	0.40	1.53	55.60
	PageRank	9.65	1.50	0.58	1.09	1.93	43.56
	Load	953.12	1.03	0.11	1.21	5.17	331.34

Table 8. Empirical runtime: PANDA-GCN forward pass duration (in milliseconds).

Method	REDDIT-BINARY	IMDB-BINARY	MUTAG	ENZYMES	PROTEINS	COLLAB
GCN	11.5	10.6	7.32	19.2	12.1	38.5
PANDA-GCN	41.3	33.9	27.3	68.4	45.5	98.7

I. Additional Experiments on Graph Classification

I.1. Relational GNNs with PANDA

Table 9 shows the experimental results for R-GCN and R-GIN. Our **PANDA** method shows the highest accuracy on most datasets for both R-GCN and R-GIN, significantly outperforming the baseline ("none") and other rewiring methods. For example, in the RGIN model, **PANDA** is shown to be the most effective method, achieving an accuracy of 91.36 on REDDIT-BINARY and 88.2 on MUTAG. Surprisingly, for ENZYMES, **PANDA** shows a significant 36.09% improvement over the baseline (None) method. In R-GCN, GTR shows higher performance than **PANDA**. However, overall, **PANDA** shows high effectiveness in R-GCN and R-GIN.

I.2. Impact of Centrality Metrics on PANDA-GIN

Table 10 compares results obtained using different kinds of centrality metrics in PANDA-GIN. For PANDA-GIN, the PageRank centrality performs best on REDDIT-BINARY, IMDB-BINARY, and MUTAG. This shows that in social networks such as REDDIT-BINARY and IMDB-BINARY, the dimension of the node that receives links from highly influential nodes must be large, as is the characteristic of PageRank.

I.3. Comparison of Centrality and Random Selection

Table 11 shows the results of an experiment by randomly expanding k nodes according to the optimal k number for each dataset. As a result of the experiment, in the case of ENZYMES, PROTEINS, and REDDIT, there are clear differences. However, in MUTAG, both results are comparable. The graph statistics of MUTAG are a minimum of 10 nodes, a maximum of 28 nodes, and an average of 17.93 nodes in a graph. Therefore, selecting some nodes randomly may not differ much from selecting nodes based on centrality. On the other hand, in the case of ENZYMES, the average number of nodes is 32.36, and the maximum number of nodes is 126, so there is a non-trivial difference between them.

Table 9. Results of our method and rewiring methods for R-GCN and R-GIN. We show the best three in red (first), blue (second), and purple (third).

Method	Reddit-Binary	IMDB-Binary	Mutag	Enzymes	Proteins	Collab
R-GCN (None)	49.850 ± 0.653	50.012 ± 0.917	69.250 ± 2.085	28.600 ± 1.186	69.518 ± 0.725	33.602 ± 1.047
+ Last Layer FA	49.800 ± 0.626	50.650 ± 0.964	70.550 ± 1.810	28.233 ± 1.138	69.527 ± 0.815	34.732 ± 1.194
+ Every Layer FA	49.950 ± 0.593	50.500 ± 0.891	70.500 ± 1.836	33.400 ± 1.142	71.670 ± 0.882	33.616 ± 0.978
+ DIGL	49.995 ± 0.619	49.670 ± 0.843	73.400 ± 2.007	28.283 ± 1.213	68.232 ± 0.851	19.926 ± 1.441
+ SDRF	58.620 ± 0.647	53.640 ± 1.043	72.300 ± 2.215	33.483 ± 1.245	69.107 ± 0.759	67.990 ± 0.386
+ FoSR	76.590 ± 0.531	64.050 ± 1.123	84.450 ± 1.517	35.633 ± 1.151	73.795 ± 0.692	70.650 ± 0.482
+ GTR	80.180 ± 0.600	65.090 ± 0.930	85.500 ± 1.470	41.330 ± 1.280	75.780 ± 0.760	74.340 ± 0.410
+ PANDA	80.200 ± 0.913	66.790 ± 1.088	90.050 ± 1.466	43.900 ± 1.176	76.000 ± 0.802	71.400 ± 0.376
R-GIN (None)	87.965 ± 0.564	68.889 ± 0.872	83.050 ± 1.439	39.017 ± 1.166	70.500 ± 0.809	75.544 ± 0.323
+ Last Layer FA	89.995 ± 0.647	69.710 ± 1.025	80.600 ± 1.639	48.183 ± 1.401	70.304 ± 0.844	75.434 ± 0.491
+ Every Layer FA	56.855 ± 0.943	71.480 ± 0.876	83.050 ± 1.518	54.950 ± 1.331	71.045 ± 0.909	75.432 ± 0.475
+ DIGL	74.425 ± 0.723	63.930 ± 0.947	81.450 ± 1.488	37.600 ± 1.198	71.312 ± 0.757	54.714 ± 0.416
+ SDRF	86.825 ± 0.523	70.210 ± 0.806	82.700 ± 1.782	39.583 ± 1.333	70.696 ± 0.815	76.480 ± 0.388
+ FoSR	89.665 ± 0.416	71.810 ± 0.880	86.150 ± 1.492	45.550 ± 1.258	74.670 ± 0.692	76.480 ± 0.390
+ GTR	90.410 ± 0.410	71.490 ± 0.930	86.100 ± 1.762	50.030 ± 1.320	75.640 ± 0.740	77.450 ± 0.390
+ PANDA	91.360 ± 0.372	72.090 ± 0.936	88.200 ± 1.513	53.100 ± 1.344	76.170 ± 0.776	77.800 ± 0.355

Table 10. Results of our method and rewiring methods for PANDA-GIN. We show the best three in red (first), blue (second), and purple (third).

$C(\mathcal{G})$	Reddit-Binary	IMDB-Binary	Mutag	Enzymes	Proteins	Collab
Degree	90.640 ± 0.414	71.900 ± 0.929	86.200 ± 2.263	37.450 ± 1.376	75.759 ± 0.856	74.950 ± 0.402
Betweenness	90.700 ± 0.378	71.390 ± 0.857	86.150 ± 1.999	44.133 ± 1.363	75.027 ± 0.871	74.644 ± 0.491
Closeness	90.080 ± 0.434	71.310 ± 0.963	88.400 ± 1.612	46.200 ± 1.410	74.741 ± 0.845	74.956 ± 0.412
PageRank	91.055 ± 0.402	72.560 ± 0.917	88.750 ± 1.570	41.483 ± 1.237	74.223 ± 0.702	74.500 ± 0.464
Load	90.625 ± 0.417	71.900 ± 0.967	86.050 ± 1.774	44.150 ± 1.455	74.812 ± 0.878	75.110 ± 0.210

Table 11. Results of an experiment by randomly expanding nodes according to the optimal number for each dataset

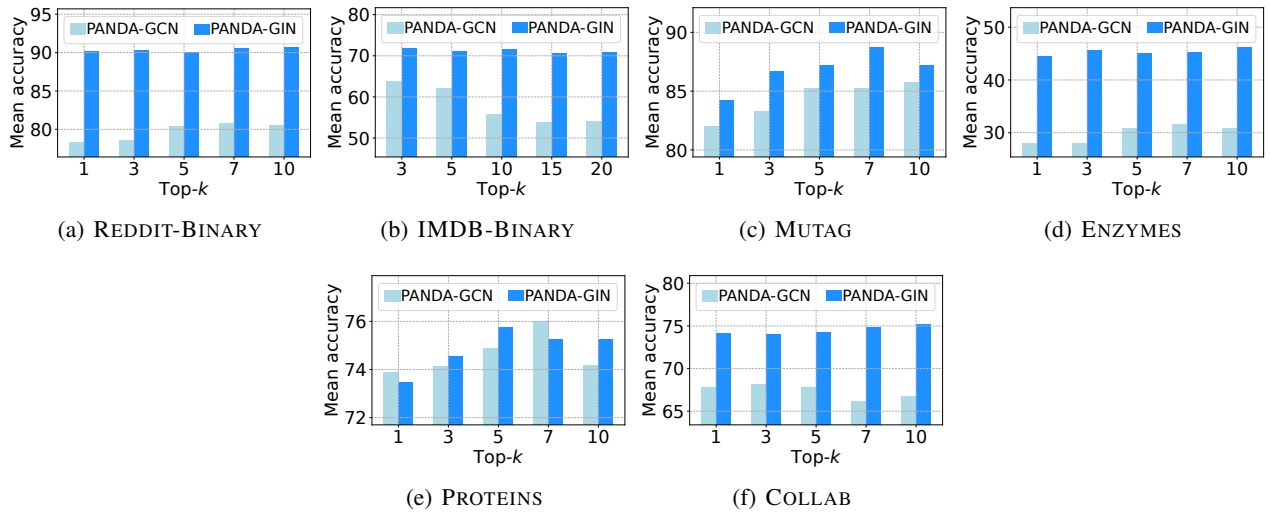| PANDA-GCN	Reddit-Binary	IMDB-Binary	Mutag	Enzymes	Proteins	Collab
Random	72.000 ± 0.850	54.000 ± 1.200	85.350 ± 1.525	26.300 ± 1.280	73.520 ± 0.795	54.435 ± 0.548
Centrality	80.690 ± 0.721	63.760 ± 1.012	85.750 ± 1.396	31.550 ± 1.230	76.000 ± 0.774	68.400 ± 0.452

J. Additional Visualizations in Section 5

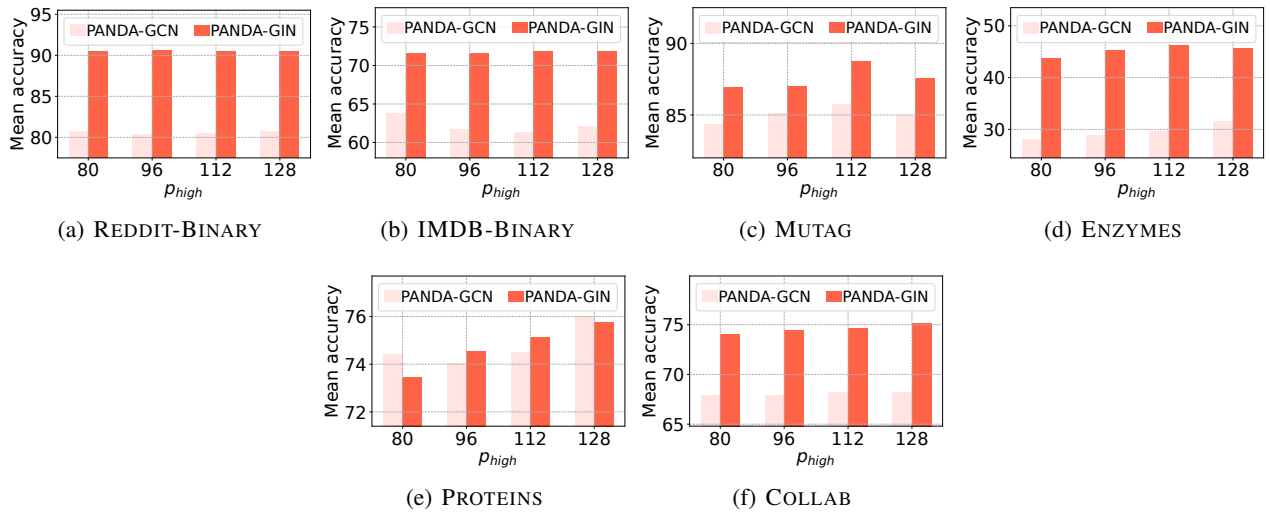Figure 13. Sensitivity on top-k for all datasets.

Figure 14. Sensitivity on p_{high} for all datasets.

K. Experiments on Node Classification

Datasets. For node classification, we use the following datasets: CORA, CITESEER (Yang et al., 2016), TEXAS, CORNELL, WISCONSIN (Pei et al., 2020), and SQUIRREL (Rozemberczki et al., 2021).

Experimental setting. Even if we allow different hidden sizes for the nodes, the output dimension of the last layer is the same for all nodes. Thus, it can also be applied to the task of node classification. For node classification, we compare **PANDA** to no graph rewiring and 4 other state-of-the-art rewiring methods: DIGL (Gasteiger et al., 2019), SDRF (Topping et al., 2022), FoSR (Karhadkar et al., 2023), and BORF (Karhadkar et al., 2023). We use the same GCN settings across all methods to rule out hyper-parameter tuning as a source of performance improvement. We accumulate results over 100 randomized trials, the same as Nguyen et al. (2023)'s setup, and report the average test accuracy with 95% confidence intervals. We use the hyperparameters reported in Table 12 as the best hyperparameters for **PANDA**.

Table 12. Best hyperparameter for node classification

Hyperparameter	TEXAS	CORNELL	WISCONSIN	SQUIRREL	CORA	CITESEER
Top-k	50	50	50	25	50	50
p_{high}	96	96	96	96	96	96
$C(\mathcal{G})$	Betweenness	Betweenness	Betweenness	Degree	Betweenness	Betweenness

Results. As shown in Table 13, in almost all cases, **PANDA** shows higher accuracy than existing rewiring methods. In cases like CORNELL and CORA, BORF still performs the best, but it is comparable to **PANDA**. **PANDA** effectively improves the classification accuracy, especially in heterophilic graphs, such as TEXAS, SQUIRREL, and WISCONSIN. In the case of WISCONSIN, it achieves higher average accuracy than both DIGL and BORF, with a 12.57% improvement over DIGL.

Table 13. Node classification accuracies of GCN with None, DIGL, SDRF, FoSR, BORF rewiring, and **PANDA** on various node classification datasets. We show the best three in red (first), blue (second), and purple (third).

Method	TEXAS	CORNELL	WISCONSIN	SQUIRREL	CORA	CITESEER
GCN (None)	44.2 ± 1.5	41.5 ± 1.8	44.6 ± 1.4	42.5 ± 2.7	86.7 ± 0.3	72.3 ± 0.3
+ DIGL	53.6 ± 1.5	44.5 ± 1.2	51.7 ± 1.3	35.2 ± 3.0	83.2 ± 0.2	74.5 ± 0.3
+ SDRF	43.9 ± 1.6	42.2 ± 1.5	46.2 ± 1.2	42.4 ± 2.5	86.3 ± 0.3	72.6 ± 0.3
+ FoSR	46.0 ± 1.6	40.2 ± 1.6	48.3 ± 1.3	42.5 ± 1.9	85.9 ± 0.3	72.3 ± 0.4
+ BORF	49.4 ± 1.2	50.8 ± 1.1	50.3 ± 0.9	42.6 ± 2.8	87.5 ± 0.2	73.8 ± 0.2
+ **PANDA**	55.4 ± 1.6	47.5 ± 1.4	58.2 ± 1.4	43.1 ± 0.9	87.0 ± 0.3	75.3 ± 0.3

L. Experiments on Long Range Graph Benchmark

Datasets. We use the Peptides (15,535 graphs) dataset from the Long Range Graph Benchmark (LRGB) (Dwivedi et al., 2022). There are two tasks related to peptides: i) PEPTIDES-FUNC, a peptide feature classification task, and ii) PEPTIDES-STRUCT, a peptide structure regression task.

Experimental setting. We compare **PANDA** to no graph rewiring and 4 other state-of-the-art rewiring methods: DIGL (Gasteiger et al., 2019), SDRF (Topping et al., 2022), FoSR (Karhadkar et al., 2023), and BORF (Karhadkar et al., 2023). On the PEPTIDE-FUNC dataset, we use two experimental setups: the same experimental setup as Nguyen et al. (2023) and the commonly used experimental setup of Dwivedi et al. (2022) for a fair comparison between our method and the rewiring method. The setting for Nguyen et al. (2023) uses a lower hidden dimension of 64 instead of 300, while the setting for Dwivedi et al. (2022) uses a hidden dimension of 300. In PEPTIDE-STRUCT dataset, we use the same experimental settings as those of Dwivedi et al. (2022) after fixing the number of layers to 5. In particular, no additional positional features are used to confirm the effectiveness of the rewiring method and the effectiveness of **PANDA**. We only replace the last layer of the backbone GCN with **PANDA** message passing for performance and efficiency.

Results. In Table 14, high average precision (AP) and low mean absolute error (MAE) values are preferred. On the PEPTIDE-FUNC dataset, for the setting of Nguyen et al. (2023), FoSR and BORF slightly improve the average precision

over GCN, but **PANDA** outperforms them. In the case of Dwivedi et al. (2022)'s setting, existing rewiring methods show trivial improvement compared to GCN, while **PANDA** improves by 1.65%. Additionally, in PEPTIDES-STRUCT, FoSR achieves a performance improvement of 0.658%, but **PANDA** achieves a performance improvement of 6.407%. This shows that the **PANDA** message passing is effective even on LRGB datasets.

Table 14. Results of GCN with None, SDRF, FoSR, BORF, and PANDA on PEPTIDES-FUNC and PEPTIDES-STRUCT. We show the best three in red (first), blue (second), and purple (third).

Method	PEPTIDES-FUNC (Test AP ↑)		PEPTIDES-STRUCT (Test MAE ↓)
	Nguyen et al. (2023)'s setting	Dwivedi et al. (2022)'s setting	Dwivedi et al. (2022)'s setting
GCN (None)	44.2 ± 1.5	59.30 ± 0.23	0.3496 ± 0.0013
+ SDRF	43.9 ± 1.6	59.47 ± 1.26	0.3478 ± 0.0013
+ FoSR	46.0 ± 1.6	59.47 ± 0.35	0.3473 ± 0.0007
+ BORF	45.6 ± 1.5	59.94 ± 0.37	0.3514 ± 0.0009
+ **PANDA**	55.4 ± 1.6	60.28 ± 0.31	0.3272 ± 0.0001