

# LEED: Local Embedding Evolution Distance for over-smoothing estimation and virtual node selection in GNN

Killian Cressant, Pedro B. Velloso, *CNAM, France*

**Abstract**—Graph Neural Networks (GNNs) suffer from two fundamental limitations: over-smoothing, where node representations become indistinguishable with depth, and over-squashing, where long-range information is compressed through limited message-passing channels. Existing metrics such as Dirichlet energy provide global characterizations of over-smoothing but lack the resolution to analyze node-level behavior and guide architectural improvements. In this paper, we propose LEED (Local Embedding Evolution Distance), a novel local metric that quantifies over-smoothing by tracking the evolution of individual node embeddings across layers. By operating at the node level, LEED enables fine-grained analysis of representation dynamics during training, revealing heterogeneous over-smoothing patterns that are invisible to global energy-based measures. This locality induces informative node importance scores, interpreted as embedding-driven centrality measures. We leverage LEED to design a more efficient strategy for virtual node selection. Unlike existing approaches that depend on multiple heuristic centrality measures, our method uses LEED as a unique criterion to guide the construction of Local Virtual Nodes to mitigate over-squashing. Experiments show that LEED provides more informative diagnostics than Dirichlet energy while preserving global evaluation, and enables more effective virtual node integration, improving GNN performance across datasets.

**Index Terms**—GNN, graph rewiring, over-smoothing, over-squashing, virtual node

## I. INTRODUCTION

GRAPH Neural Networks (GNN) have been developed in the past years to leverage structured topological data to enhance the performance of neural network models. They have been successfully applied across a wide range of domains, including networking applications [1], [2], where network topology naturally defines the graph structure, molecular datasets such as classical ENZYMES and MUTAG, citation networks with CORA dataset [3] and even traditional computer vision tasks [4]. Despite its recent success, GNNs still face fundamental challenges specially regarding the influence of the message-passing process on the quality and expressiveness of node embeddings [5].

In this context, one important issue in GNNs is the over-smoothing problem, which makes node embeddings indistinguishable as information is propagated across the graph, leading to a severe performance degradation [6]. Another important challenge is the over-squashing problem [7]. It occurs specially

when messages from distant nodes are aggregated as they propagate across graph layers, causing structurally important information to be compressed, distorted, or discarded. This problem is intensified in the presence of bottleneck nodes and limited-capacity embeddings.

Previous studies have shown that using too many layers in GNN architectures inevitably results in over-smoothing even if it is necessary to pass long-range dependencies. To mitigate this issue, Kipf and Welling proposed using a shallow two-layer GCN [3]. Subsequent works, such as [8], further demonstrated that the GCN model can be simplified by removing intermediate nonlinear activation functions without compromising its performance. This phenomenon relating the number of layers to the loss of expressive power was later studied in [9]. In addition, different approaches have been developed to combat over-smoothing [10]. In particular, researchers have proposed to modify the global loss function by incorporating regularization terms [11]. In this direction, a prominent approach involves the use of Dirichlet energy [12]. By monitoring or regularizing this energy, one can prevent the signal from collapsing into a uniform state.

More recently, the focus has shifted toward graph rewiring [13]. This technique relies on the fact that the original graph structure, often dictated by physical or logical relationships in the data, is not necessarily the optimal topology for training GNNs [14]. Therefore, graph rewiring techniques dynamically [15] or pre-emptively alter the edge or node set to improve message flow, effectively bypassing bottlenecks to reduce over-squashing [16] or pruning redundant paths to prevent over-smoothing. A central challenge in these frameworks lies in identifying the critical nodes that would most benefit from the addition of local transformations. However, existing graph rewiring approaches largely rely on empirically testing a range of classical centrality measures, such as degree, betweenness or PageRank, and selecting the one that yields the best performance.

In this paper we propose LEED (Local Embedding Evolution Distance), a node-level metric designed to characterize over-smoothing by monitoring how individual node representations change across network layers. Therefore, the main advantage of LEED consists of addressing both the over-smoothing and over-squashing problems simultaneously. By focusing on per-node dynamics, LEED captures fine-grained variations in representation collapse that global energy-based indicators fail to detect. In addition, we exploit its local properties to replace all the classical centrality metrics to

Killian Cressant and Pedro B. Velloso are from the CEDRIC Lab, Conservatoire National des Arts et Métiers (Cnam), 75003 Paris, France (email: killian.cressant@cnam.fr; pedro.velloso@cnam.fr).

find the critical nodes for graph rewiring. Therefore, the centrality metric is no longer a parameter of the model. Our experimental results demonstrate that LEED offers more precise and informative diagnostics of over-smoothing than Dirichlet energy while maintaining comparable global assessment capabilities. Furthermore, we apply LEED as a unique metric to guide two different graph rewiring solutions to select the critical nodes. Finally, we show that LEED improves GNN performance across six classical datasets [17] with different characteristics by mitigating over-squashing, and, at the same time, controlling over-smoothing.

The remainder of this paper is organized as follows. In Section II, we present our preliminary notation related to GNNs, as well as the basic aspects of graph rewiring. Section III presents the related work concerning the over-smoothing estimation and critical nodes selection in graph rewiring. In Section IV we introduce LEED along with its design issues and its definitions. Section V we define practical usage of critical nodes. In Section VI and VII, we present our experiments and the main results. Finally, Section VIII presents our conclusions and future work.

## II. BACKGROUND

Let  $G = (V, E)$  be a undirected graph with a set of node  $V$  and a set of edges  $E$ . We consider the adjacency matrix of the graph  $\mathbf{A}$ , the neighborhood of a node  $i$  is  $N_i = \{j \in V | (j, i) \in E\}$  and the embedding input features is  $\mathbf{X} = (X_1, \dots, X_n)$ , with  $n = |V|$ . Then, the general equation of a layer of a GNN (working with message-passing layer) can be written as:

$$X_i^{(k+1)} = \text{Update}^{(k)}(X_i^{(k)}, \text{Agg}^{(k)}(\{X_j^{(k)}, j \in N_i\}))$$

If we consider the GCN of Kipf and Welling [3], we obtain:

$$\mathbf{X}^{(k+1)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{X}^{(k)} \mathbf{W}^{(k)})$$

with  $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ ,  $\mathbf{A}$  the adjacency matrix of the graph and  $\mathbf{I}_N$  the identity matrix,  $\tilde{\mathbf{D}}$  the diagonal matrix elements of  $\tilde{\mathbf{A}}$ ,  $\mathbf{W}^{(k)}$  the trainable weight matrix of the GNN and  $\sigma$  an activation function, such as ReLU. This equation comes from the initial Graph Convolutional Networks (GCN) paper [3], where they develop a GNN based on graph convolution. The main idea is to write the convolution in the Fourier domain, then use a Chebyshev polynomial approximation of the eigenbase of the Laplacian  $\Delta = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$  to finally construct the layer operation, creating a first-order spectral filter layer, followed by an activation function. The activation function is the update function and the aggregation is made by convolution. Since GCNs inherit a strong influence from the GNN message-passing paradigm, this equation remains quite accurate for most GNN models, with the matrix formulation preserved and only minor modifications to the update rule [8] or to the aggregation mechanism, as in GAT [18].

### A. Over-smoothing vs over-squashing

In the last years, these two phenomena have been observed in GNNs and have attracted significant attention in the literature [6], [7], [19]. The basic definitions for these two problems are as follows:

- **Over-smoothing:** a result of repeated aggregations that cause node features to converge to a stationary point, effectively washing out the local signals necessary for downstream tasks.
- **Over-squashing:** this occurs when a structural bottleneck (such as a bridge or “bottleneck” node) is forced to compress a disproportionately large volume of information from one subgraph to another. This forced compression leads to a significant loss of information, preventing long-range dependencies from being effectively captured.

While over-smoothing typically arises from excessive message-passing across densely connected structures, leading to indistinguishable node representations, over-squashing occurs under the opposite topological conditions. As a consequence, a trade-off emerges when attempting to mitigate both phenomena simultaneously [20]. Hence, in general, these two problems are addressed separately in the literature.

Over-smoothing is usually handled through regularization or GNN layer modifications. Graph rewiring, on the other hand, is predominantly used to alleviate over-squashing, and the most common techniques include:

- Adding virtual nodes [21]. It consists of adding one [22] or several central virtual nodes [23] to the original graph. As the over-squashing problem intensifies when the commute time between nodes is high, the key idea is to reduce drastically and efficiently the graph diameter, and as a consequence, the commute time of the graph.
- Using topological curvature [24], such as Ricci curvature [25], Olivier curvature or similar measures. In this case, the main goal is to add edges in specific regions of the graph that act as topological bottlenecks, thereby alleviating structural constraints.
- Lastly, some works focus on expander graphs. Expander graph is a family of graphs that has a high Cheeger constant, which can be associated with fewer topological bottlenecks. In this setting, virtual nodes and topological modifications are combined to construct an entirely new graph [26]. These approaches share the same theoretical motivation as curvature-based methods: modifying or removing all bottleneck from the graph structure will improve the modeling of long-range dependencies.

### B. Graph rewiring with virtual nodes

The use of graph rewiring techniques is inherently linked to computational efficiency considerations. Even though it is possible to design increasingly expressive message passing architectures, as for instance by enhancing aggregation and update functions to enable information propagation across distant regions of a graph within a limited number of layers, as exemplified by attention-based models, such as Graph Attention Networks (GATs [18]), such approaches often incur substantial computational overhead. This overhead can significantly limit their scalability and practicality on large-scale graphs. Moreover, even these advanced GNN architectures can benefit from graph rewiring strategies, suggesting that architectural expressiveness alone does not fully address structural limitations in message-passing. Consequently, in

many practical settings, simpler architectures such as GCNs or closely related variants are preferred due to their favorable trade-off between expressiveness and computational efficiency, and because their performance can be effectively enhanced through appropriate graph rewiring techniques.

The earliest approach to mitigating over-squashing through virtual nodes introduced a single, globally connected virtual node [22]. This construction effectively reduces the graph diameter to at most two, enabling rapid information propagation across distant nodes. While such methods yield encouraging results on certain datasets, their effectiveness is limited in large or complex graphs, as a single virtual node must encode global information within a constrained representation space. To address this limitation, subsequent works have proposed the introduction of multiple virtual nodes [27]. By increasing the number of virtual nodes, these methods improve representational capacity and extend applicability to more challenging datasets. However, this comes at the cost of increased computational complexity, as each additional virtual node introduces new edges that substantially raise the message-passing overhead. Furthermore, these approaches significantly alter the original graph topology, which may not be desirable for all tasks.

The impact of such topological modifications appears to be dataset-dependent [28]. Graph rewiring and virtual node techniques have shown strong performance on benchmark graph classification datasets such as MUTAG and PROTEINS, where task-relevant information is largely encoded in the graph structure itself. In contrast, for citation networks such as CORA and CITESEER, node attributes play a dominant role and the underlying topology is less directly correlated with the target labels. In these datasets, aggressive graph rewiring often leads to degraded performance [29]. This discrepancy suggests that the effectiveness of virtual nodes and rewiring strategies depend on the extent to which task-relevant information is inherently topological, rather than solely relational or feature-driven. To address this problem, the authors in [29] propose the use of multiple Local Virtual Nodes (LVN) rather than a single global virtual node. This approach aims to limit drastic modifications to the graph topology, as depicted in Figure 1, while capturing more localized information and improving the training dynamics of the virtual nodes themselves. While this approach mitigates some of the drawbacks of global virtual nodes, it introduces additional complexity during training, including structural adaptations such as directional virtual nodes, which can substantially increase computational cost.

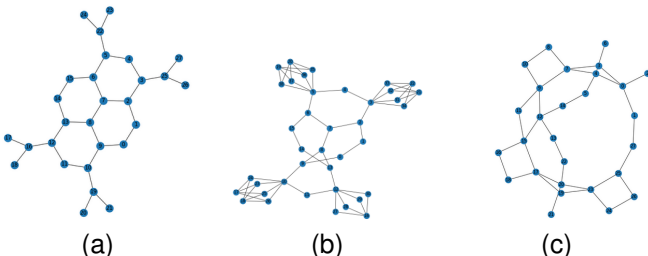


Fig. 1. (a) The original graph; (b) The graph generated by LVN approach; (c) The graph generated using the Expander graph approach

In this context, PANDA [30], introduces an approach conceptually similar to virtual nodes by modifying the width of selected nodes, namely, by changing their embedding dimensionality. Intuitively, this mechanism resembles the local virtual nodes introduced in [29], as both methods aim to identify central nodes in order to improve local message-passing efficiency by locally increasing the capacity for information storage. However, this approach requires modifying the message-passing functions to accommodate embeddings of multiple dimensionalities. As a consequence, it incurs a higher computational cost than LVN in the undirected setting, while achieving comparable performance.

### III. RELATED WORK

Several works address the over-smoothing estimation problem [31], while other works focus on over-squashing problem [7], and analyse and apply different centrality metrics to define the importance of nodes for graph rewiring [29]. Some studies explicitly monitor over-smoothing while mitigating over-squashing [20], [30], but in most cases these phenomena are treated independently, even when relying on similar theoretical tools, such as spectral analysis [13]. In this work, we propose LEED, that takes a first step toward jointly addressing both effects.

#### A. Over-smoothing estimation

Most techniques used to evaluate over-smoothing in GNNs rely on quantities equivalent or closely related to the Dirichlet energy [6], [12], which is computationally efficient and provides interesting results on over-smoothing. When this energy becomes low, node embeddings tend to become indistinguishable, resulting in a loss of discriminative information. Conversely, higher Dirichlet energy values correspond to more distinguishable node representations. However, there is no easy way to describe local over-smoothing effect. In [19], the authors introduce MAD, an over-smoothing metric grounded in topological properties. While MAD computes cosine similarities between node embeddings on a per-node basis, it relies on a graph-wide filtering mechanism and restricts its analysis to direct neighbors. As a consequence, this formulation is limited in its ability to capture bridge or bottleneck structures, which typically involve interactions beyond immediate neighborhoods. In addition, the use of a global filtering operation reduces the locality of the resulting metric.

To address these limitations, LEED incorporates a corrected two-hop neighborhood aggregation centered around each node. This design enables LEED to capture richer local structural information, including nodes involved in long-range information transfer, while maintaining consistency with global over-smoothing trends. In summary, LEED is more efficient at capturing local properties, while exhibiting graph-level behavior comparable to Dirichlet energy.

#### B. Finding critical nodes

A central challenge in these frameworks lies in identifying the nodes that would most benefit from the addition of local

virtual nodes, like LVN approaches or adjustments to their embedding width, like PANDA. This problem remains largely unresolved. As a consequence, to address this issue, these frameworks evaluate several classical centrality measures, such as PageRank, degree, betweenness, and closeness, to empirically select the most effective one for a given task. However, this strategy requires testing multiple centrality measures and does not provide a principled criterion for their selection. In addition, prior theoretical analyses suggest that mitigating over-squashing with graph rewiring is more effectively achieved by strengthening bottleneck or bridge structures rather than emphasizing globally central nodes [20]. This theoretical mismatch highlights the need for alternative criteria that are more directly aligned with the mechanisms underlying over-squashing in GNNs.

In [21], the authors observe that the benefits of virtual nodes are not directly associated with mitigating over-smoothing and may, in some cases, exacerbate it. Nevertheless, over-smoothing is not inherently detrimental and can be advantageous for graph-level tasks, where homogenized node representations facilitate global aggregation. This perspective provides a plausible explanation for the contrasting performance of GNNs augmented with virtual nodes on node-level benchmarks such as CORA and CITESEER, compared to graph-level classification datasets such as MUTAG. These observations are consistent with the intuition that introducing virtual nodes increases graph connectivity and edge density, which can accelerate the over-smoothing process. While this effect may hinder node-level discrimination, it can be beneficial for tasks that require global structural awareness.

The Local Virtual Node (LVN) framework represents a promising direction by offering flexible mechanisms for localized information aggregation. However, its practical deployment is hindered by the need to tune multiple hyperparameters, including the choice of centrality measure, the number of central nodes considered, and the number of virtual nodes introduced per selected node. In addition, the framework inherits the theoretical limitations associated with classical centrality-based selection and lacks guarantees regarding its impact on training dynamics. Rather than introducing an entirely new framework with similar limitations, we build upon and refine the LVN paradigm by proposing LEED, a novel metric specifically designed to improve training dynamics in GNNs. Our approach eliminates the need for dataset-specific centrality selection while directly addressing the structural challenges that arise during message-passing.

#### IV. LEED

LEED (Local Embedding Evolution Distance), is a node-level metric designed to characterize over-smoothing in GNNs. The basic idea is to quantify the evolution of node embeddings over the message-passing process to understand how individual node representations change across network layers. In addition, LEED can be used as a unified metric to find critical nodes in graph rewiring strategies aimed at alleviating over-squashing. As a result, LEED offers the key advantage of jointly addressing both over-smoothing and over-squashing problems within a single framework.

##### A. Design principles

Ideally, any local over-smoothing evaluation should take the form of a distance measure [32], yielding symmetric values in  $\mathbb{R}^+$ . At the graph level, the notion of symmetry is not directly meaningful; however, at the node level, any measure quantifying the dissimilarity between two nodes should be symmetric. These local distances can then be aggregated to construct a global metric for the entire graph.

Another important aspect to consider is the trade-off between over-smoothing and over-squashing, as mentioned in Section II-A. These two phenomena are often antagonistic and therefore closely related. As a consequence, adding edges or nodes can alleviate over-squashing by reducing topological bottlenecks in the graph; however, the same modifications may simultaneously exacerbate over-smoothing by increasing connectivity between previously weakly connected regions of the graph. Therefore, by combining solutions designed to address over-smoothing with methods that mitigate over-squashing, it is possible to reduce the risk of trading one problem for the other.

Hence, in this work, we build upon the Dirichlet energy [12] and adapt it to our setting by focusing on the identification of critical nodes, since central nodes are as important as bridge nodes in graph rewiring [20]. We therefore refer to these nodes as *critical nodes*, a notion that we define and discuss in detail in the following sections.

The classical Dirichlet energy of an embedding  $X^{(k)}$  of a GCN can be calculated by:

$$E(X^{(k)}) = \text{tr}\left(X^{(k)\top} \tilde{\Delta} X^{(k)}\right) \quad (1)$$

$$= \frac{1}{2} \sum_{i,j} a_{ij} \left\| \frac{x_i^{(k)}}{\sqrt{1+d_i}} - \frac{x_j^{(k)}}{\sqrt{1+d_j}} \right\|_2^2 \quad (2)$$

with  $\tilde{\Delta}$  the Laplacian modified operator of the graph,  $a_{i,j}$  the adjacency value in  $i, j$  and  $d_i$  the degree of the adjacency matrix for node  $i$ . This formula is computationally efficient and provides interesting results on over-smoothing. However, there is no easy way to describe local over-smoothing effect. We can indeed reduce the embedding  $X$  to a specific portion of the signal, but the meaning of the Dirichlet energy is lost in this computation. The over-smoothing phenomenon results from both the GCN architecture and the graph structure. As GCN architectures become deeper, node embeddings tend to become increasingly similar. The same effect is also observed in highly connected graphs. The combination of both factors can amplify over-smoothing.

Therefore, in this paper, rather than viewing over-smoothing as a global optimization concern as is commonly done through regularization terms based on Dirichlet energy, we argue that over-smoothing should not be viewed solely as a message-passing issue, but more fundamentally as a graph connectivity problem: nodes belonging to dense substructures, such as cliques, tend to exhibit embeddings that are significantly more similar to each other than to the rest of the graph. Figure 2 illustrates this effect using one embedding dimension for a clique and its immediate neighborhood after GNN training

(two-layer GCN). Nodes within the clique exhibit highly similar embeddings, whereas neighboring nodes can have substantially different representations. Based on our empirical analysis on the CORA dataset, embeddings of nodes within a clique are at least twice as close to each other as to those of their direct neighbors. Moreover, as the number of neighboring nodes decreases, this similarity further increases, reaching up to five times that observed in more diffuse graph regions. In such cases, the absolute value of the Dirichlet energy is not the main concern; rather, the critical issue lies within the clique itself.

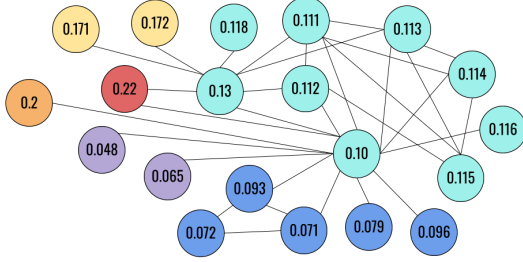


Fig. 2. One dimension of node embedding around a clique

Consequently, addressing over-squashing and over-smoothing requires interventions both at the message-passing level and at the level of the graph structure [33]. While many GNN models have been proposed to mitigate these issues, far fewer approaches focus on improving the graph structure for GNN training, and even fewer consider both over-squashing and over-smoothing simultaneously. This gap is partly due to the scarcity of methods capable of performing local evaluations. As with Dirichlet energy, most existing tools provide global solutions, even though the graph structure can and should be treated locally.

### B. Local Embedding Evolution Distance

The goal of this work is to propose a tool to enhance GNN performance by addressing both over-smoothing and over-squashing through local graph structure modifications. Therefore, our method enables analysis of the local evolution of node embeddings depending on the underlying graph structure, as illustrated in Figure 3. LEED assigns a local score  $l_i$  to each node  $i$  based on the 2-hop embedding evolution. Aggregating these scores over all nodes yields a global measure that is comparable to Dirichlet energy, while considering the full set of node-wise scores produces a matrix that allows direct comparison of node importance with respect to the overall over-smoothing, as can be observed in the heatmap.

One way to assess whether a node embedding is close to its neighborhood is to examine how similar its embedding vector is to those of its neighbors. In the Dirichlet energy, this notion is captured by aggregating embedding differences over all neighboring node pairs. In contrast, LEED focuses on local proximity by considering only the minimum distance between a node and its neighbors, rather than summing distances across all edges. Furthermore, instead of directly comparing node embeddings, we apply a transformation prior to distance

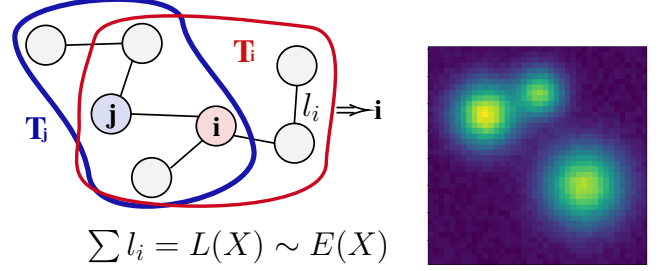


Fig. 3. LEED score and the Heatmap highlighting critical nodes

computation. The Dirichlet distance can be interpreted through the lens of a Markov diffusion kernel [34]. In that work, the authors define the Markov diffusion distance as:

$$d_{i,j}(K) = \|x_i(K) - x_j(K)\|_2^2 \quad (3)$$

With  $K$  a time parameter, which can be seen as analogous to the GNN layer index  $k$ , although it is not equivalent. That equation leads to a formulation closely related to Dirichlet energy regularization. This connection highlights the close relationship between Markov processes and GCNs, demonstrating that the two are inherently linked. Following this perspective, we propose a Markov evolution process designed to mimic the embedding dynamics of GCNs. Moreover, rather than considering all neighboring pairs as in Dirichlet energy, we retain only the smallest local distance by applying a minimum operator. While the overall computational cost remains comparable (see in VII-A for details), the algebraic formulation is not straightforward. Nevertheless, using a minimum instead of a sum is more appropriate for local distance evaluation: aggregation is meaningful at the graph scale, whereas the minimum provides a sharper characterization of local embedding variations. We define the distance as:

$$\ell(x_i^{(k)}) = \min_{j \in \mathcal{N}_i} \|T_m(x_i^{(k)}) - T_m(x_j^{(k)})\|_2^2 \quad (4)$$

where  $T_m$  is a mean evolution function, depending of  $m$ -hops, weighted by the adjacency matrix where we included a correction for the self-loop:

$$T_1(x_i^{(k)}) = \frac{\sum_{p \in \mathcal{N}_i} a_{ip} x_p^{(k)} + x_i^{(k)}}{\sum_{p \in \mathcal{N}_i} a_{ip} + 1} \quad (5)$$

and the idea is to get close to such equation for 2-hops:

$$T_2(x_i) = \frac{\sum_{p \in \mathcal{N}_i} a_{i,p} T_1(x_p) + 2T_1(x_i)}{\sum_{p \in \mathcal{N}_i} a_{i,p} + 2} \quad (6)$$

However, since node embeddings are not always available, as in the case of the COLLAB dataset, we modify  $T_2$  to depend solely on node degree. In this setting, we obtain Equation 7.

$$T_2(x_i^{(k)}) = \frac{\sum_p a_{ip} \left( \frac{\sum_q a_{pq} x_q^{(k)} \cdot 2 \cdot (\delta_{i,q} + 1)}{\sum_q a_{pq} + 2} \right)}{\sum_p a_{ip} + 1} \quad (7)$$

$\delta_{i,j}$  denotes the Kronecker symbol, which is used to assign more weight to self-loops during the process. In practice, we do not consider more than two-hop neighborhoods. Following [34], we extend this formulation to the entire graph. by summing over all node embeddings, resulting in:

$$L(X^{(k)}) = \sum_{i=1}^n \ell(x_i^{(k)}) \quad (8)$$

At the graph level, the Dirichlet energy  $E(X^{(k)})$  exhibits desirable properties that we aim to preserve. In particular, we can derive lower and upper bounds of  $E(X^{(k)})$  based on the representations at the previous layer:

$$0 \leq E(X^{(k)}) \leq s_{\max}^{(k)} E(X^{(k-1)}), \quad (9)$$

with  $s_{\max}^{(k)}$  the squares of maximum singular values of  $W^{(k)}$ . In the original papers, this constraint is used to define two hyperparameters  $c_{\min}$  and  $c_{\max}$  to ensure that the Dirichlet energy remains within a suitable range during training:

$$c_{\min} E(X^{(k-1)}) \leq E(X^{(k)}) \leq c_{\max} E(X^{(0)}) \quad (10)$$

Since we aim to preserve this property, we seek either to derive our own bounds or to find a link between  $E(X^{(k)})$  and  $L(X^{(k)})$ . This approach leads, in practice, to the following result:

$$L(X^{(k)}) \leq \left( \max_{i \in N} \frac{2}{d_i} \right) C_{\hat{T}} E(X^{(k)}), \quad (11)$$

with  $C_{\hat{T}}$  a value depending of  $T$  and  $X$ .

**Proof.** There are two steps in this proof.

Let  $G = (V, E)$  be an undirected graph with  $|V| = n$ , adjacency matrix  $A = (a_{ij})_{1 \leq i, j \leq n}$ , and degrees

$$d_i := \sum_{j=1}^n a_{ij}$$

Let  $x_i^{(k)} \in \mathbb{R}^d$  be the feature vector of node  $i$  at layer  $k$  and

$$X^{(k)} := (x_1^{(k)}, \dots, x_n^{(k)}) \in (\mathbb{R}^d)^n$$

**Step 1: Comparison with unnormalized dirichlet energy.** For each fixed  $i$ ,  $T_2(x_i^{(k)})$  is a finite linear combination of the vectors  $\{x_q^{(k)}\}_{q=1}^n$  with coefficients depending only on  $A$ . Hence, there exists a matrix  $V = (V_{iq}) \in \mathbb{R}^{n \times n}$  such that

$$T_2(X^{(k)}) = V X^{(k)}$$

Since  $\mathcal{N}_i$  is finite and nonempty for all  $i$ , For

$$S_i := \{\|T_2(x_i) - T_2(x_j)\|_2^2, j \in \mathcal{N}_i\}$$

We have:

$$\min S_i \leq \overline{S_i} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} a_j \|T_2(x_i) - T_2(x_j)\|_2^2$$

Applying this inequality for a symmetric binary matrix, and summing over nodes ( $d_i = |\mathcal{N}_i|$ ) we get:

$$L(X^{(k)}) \leq \sum_{i=1}^n \frac{1}{d_i} \sum_{j=1}^n a_{ij} \|T_2(x_i^{(k)}) - T_2(x_j^{(k)})\|_2^2 \quad (\star)$$

Define the (unnormalized) Dirichlet energy of a signal  $Z = (z_1, \dots, z_n)$  by

$$\mathcal{E}(Z) := \frac{1}{2} \sum_{i,j=1}^n a_{ij} \|z_i - z_j\|_2^2$$

From  $(\star)$  we obtain

$$L(X^{(k)}) \leq \left( \max_{i \in N} \frac{2}{d_i} \right) \mathcal{E}(T_2(X^{(k)}))$$

This step explain why the profile of the function has good reason to be similar to the Dirichlet energy function. However, we are using a normalized Dirichlet energy, and that normalization explain the difference of range in the values proposed:

$$\begin{aligned} E(X^{(k)}) &:= \frac{1}{2} \sum_{i,j=1}^n a_{ij} \left\| \frac{x_i^{(k)}}{\sqrt{1+d_i}} - \frac{x_j^{(k)}}{\sqrt{1+d_j}} \right\|_2^2 \\ &= \mathcal{E}(\tilde{D}^{-1/2} X^{(k)}) \end{aligned}$$

**Step 2: Extension using conjugated operator.** Define the linear operator

$$\hat{T} := \tilde{D}^{-1/2} V \tilde{D}^{1/2}$$

Then

$$\mathcal{E}(T_2(X^{(k)})) = \mathcal{E}(\hat{T} \tilde{D}^{-1/2} X^{(k)})$$

Let

$$C_{\hat{T}} := \sup_{Y \notin \ker(\Delta)} \frac{\mathcal{E}(\hat{T}Y)}{\mathcal{E}(Y)} = \sup_{Y \notin \ker(\Delta)} \frac{\text{Tr}(Y^\top \hat{T}^\top \Delta \hat{T} Y)}{\text{Tr}(Y^\top \Delta Y)}$$

Applying this with  $Y = \tilde{D}^{-1/2} X^{(k)}$  yields

$$L(X^{(k)}) \leq \left( \max_{i \in N} \frac{2}{d_i} \right) C_{\hat{T}} E(X^{(k)}),$$

with for most graphs:

$$\left( \max_{i \in N} \frac{2}{d_i} \right) = 2$$

□

As some nodes (e.g., leaf nodes) are not taken into account in the computation of  $T_2$ , we cannot obtain similarly infimum. However, with all simulations we can easily see that in practice  $\exists C$  s.t  $C \cdot E(X^{(k)}) \leq L(X^{(k)})$ , leading to equivalent measurement for  $E$  and  $L$  except in special cases.

In practice  $C_{\hat{T}}$  can become very large for large-scale datasets, as illustrated in Figures 5 and 6. However, results also suggest that the profile of Dirichlet energy is likely to be similar to that of the LEED distance. Since the relative evolution of LEED is more informative than its absolute value, we design an experiment to validate this behavior. Specifically, we conduct a series of comparative experiments using eight different adjacency matrices for GNN training, and analyze whether the trajectories of Dirichlet energy and LEED over training epochs exhibit similar trends. The results are presented in Figure 4. We establish a link between these two functions, which can be exploited to impose similar constraints during

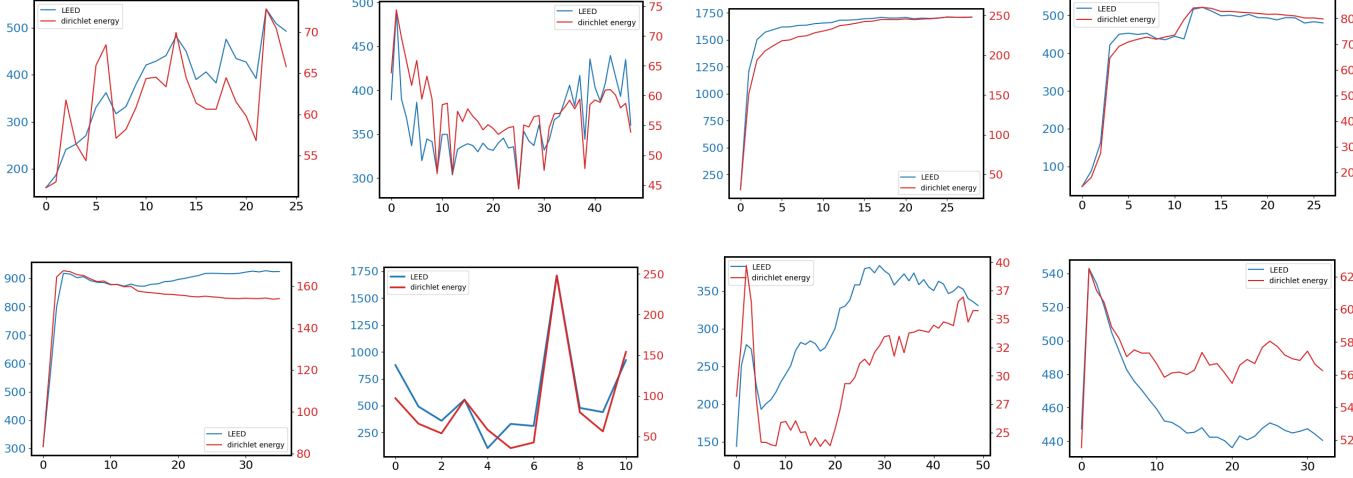


Fig. 4. Comparison with 8 configurations between LEED and Dirichlet energy

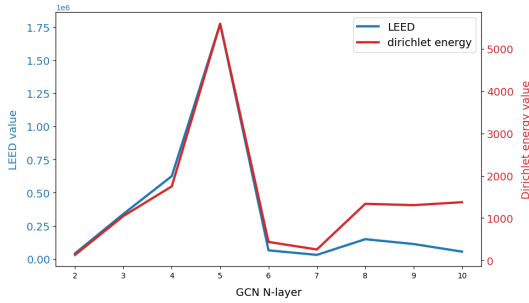


Fig. 5. Cora distance for each layer between 2 and 10

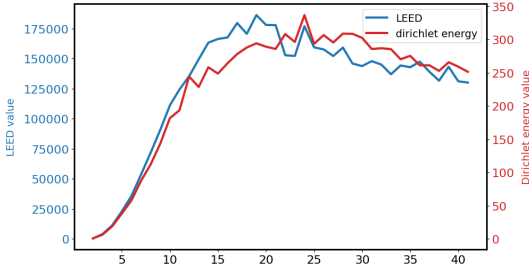


Fig. 6. Cora distance for every 10 epoch of training

training to prevent over-smoothing. We perform additional experiments to study the evolution of this distance as a function of the number of GCN layers, evaluating both LEED and the Dirichlet energy at the final layer. Results are reported in Figure 5. As the CORA dataset is not connected, the over-smoothing metrics are not expected to converge to zero. In addition, the resulting function presents local dependencies, which allows us to assess how individual nodes locally affect its score. Further investigation of these properties is left for future work.

## V. CRITICAL NODE SELECTION

An important advantage of LEED is its ability to identify critical nodes for GNN performance. We define critical nodes

as those that play a decisive role in information propagation and representation learning within the graph. This notion is motivated by prior work showing that node importance, often formalized through centrality measures, is instrumental in mitigating over-squashing [29], while recent theoretical analyses emphasize the role of bottleneck structures in limiting long-range information flow [35].

To reconcile these perspectives, we design a critical node score that jointly captures centrality-related effects, bottleneck sensitivity, and embedding distance evaluation. Specifically, the proposed score integrates local structural influence with the behavior of node embeddings during training, allowing it to reflect both topological and representational constraints. To this end, we reuse the local function  $T$ , previously introduced, and apply it in a comparative setting by incorporating input embedding values as a reference: Let

$$D_i = \|T_2(x_i^{(0)}) - x_i^{(0)}\|_2, \quad i \in [1, n] \quad (12)$$

As we do not need to search among each layers anymore, we instead use only input values and initial embeddings. Then, we obtain a set of critical nodes by taking a top-K, which is denoted by  $\tau_k$  the  $(1-k)$ -quantile of the set  $\{D_i\}_{i=1}^n$ , defined as:

$$\tau_k = \inf \left\{ t \in \mathbb{R} : \frac{1}{n} \#\{i \in [1, n] \mid D_i \geq t\} \leq k \right\} \quad (13)$$

Hence, the ensemble of critical nodes corresponding to the top  $k = \frac{X}{100}$  fraction of the  $D_i$  is given by

$$S_k = \{i \in [1, n] \mid D_i \geq \tau_k\} \quad (14)$$

### A. Properties on critical nodes

Our formulation assumes the availability of node embeddings. However, certain benchmark datasets, such as REDDIT-BINARY and COLLAB, do not provide explicit node features. In such cases, we adopt a standard strategy commonly used in graph rewiring methods [26], [30] that assigns a constant feature vector  $\mathbf{1}_n$  to all nodes. This choice corresponds to initializing each node with an identical scalar embedding. In

practice, we restrict the embedding to a single dimension, as scores computed across different embedding dimensions differ only by a multiplicative factor. For undirected graphs with a symmetric binary adjacency matrix, this simplification allows the proposed measure to be expressed in closed form, yielding the following output:

$$D_i = \frac{|d_i - 2|}{d_i + 2}, \quad (15)$$

except when  $d_p \ll d_i, \forall p \in \mathcal{N}_i$ . This simple formula yields a value in  $[0, 1]$ , that increases monotonically with  $d_i$ , the degree of node  $i$ . When seeking for solutions involving more complex embedding such as one-hot encoding of dimension  $d$ , present in MUTAG, ENZYMES datasets, we obtain:

$$\mathbb{E}[D_i^2] = \alpha_i^2 + \left(1 - \frac{1}{d}\right) \sum_{q \neq i} \beta_q^2 + \frac{2\alpha_i}{d} \sum_{q \neq i} \beta_q + \frac{1}{d} \left( \sum_{q \neq i} \beta_q \right)^2$$

with

$$\alpha_i = \frac{4}{d_i + 1} \sum_{p \in \mathcal{N}_i} \frac{1}{d_p + 1} - 1$$

and

$$\beta_q = \frac{2}{d_i + 1} \sum_{p \in \mathcal{N}(i)} \frac{\mathbb{1}_{\{q \in \mathcal{N}(p) \setminus \{i\}\}}}{d_p + 2}$$

This suggests that the score for  $D_i$  is high when node  $i$  has a limited number of neighbors  $p_i$ , but each of these neighbors has several neighbors  $q_p$  ( $\alpha$  parameter). Additionally, the value of  $\beta_q$  tends to be higher when there is no connection between any pair of  $p$ -nodes. As a result, the optimal score for node  $i$  arises when it serves to bridge distinct components of the graph. Consequently, the distance measure places greater emphasis on bridging structures, similar to curvature-based methods [13]. Hence, LEED inherits the well-established advantages for training GNNs. However, this solution applies only for a normal random distribution of the one-hot embedding. If the distribution shifts, a new parameter emerges, representing the distance between the embedding and its direct neighbors. This results in a different type of centrality measure, one that is more closely related to information importance than to topological importance [32].

## VI. EXPERIMENTAL IMPLEMENTATION

In our experimental evaluation, we compare LEED to a standalone GCN model and other three graph rewiring approaches: Cayley graph [26], LVN [29], and PANDA [30]. For their implementation, we rely on open-source code that are publicly available in github. For benchmarking, we evaluate their performance with six widely adopted datasets from the TUDataset collection [17]: MUTAG, ENZYMES, PROTEINS, REDDIT-BINARY, IMDB-BINARY, and COLLAB. These datasets are commonly used to evaluate a model's ability to capture long-range interactions in graph-level tasks. We split the datasets as 60% for the training set, 20% for the validation set and 20% for the test set.

Results for the standalone GCN model are taken directly from the Cayley graph paper, with the exception of PROTEINS, where improved performance is reported under the LVN framework and is not reproduced in our experiments. All other results are reproduced using dataset-specific hyperparameter configurations. For Cayley graph rewiring, we use the results reported in the original paper as baselines, as the full set of hyperparameters is not provided. Nevertheless, we verified that comparable performance can be achieved across all datasets under similar experimental conditions. For LVN, we extensively evaluated the framework using the hyperparameters provided by the authors, conducting 20 independent runs for reproducibility. Given the lower variance reported in the original results, we retain their reported scores. Finally, for PANDA, we employ the original framework and use the same hyperparameter search space as described by the authors, with 30 runs for MUTAG. For the remaining datasets, the search space is reduced based on the best configurations reported in the original work. We observe improved performance on two datasets and report, for each case, the best result between our experiments and those from the original paper.

To evaluate the effectiveness of LEED in selecting critical nodes for graph rewiring, we apply it to LVN [29] and PANDA [30]. In the original methods, several centrality metrics were used to determine central nodes. Our objective here is to replace all of these metrics with our proposed local distance measure. We keep all the different hyperparameters of the original frameworks. For example, both methods employ a top-K strategy to obtain the set of central nodes, and a number called 'worker' to create multiple virtual nodes for each central node selected. Similarly, PANDA uses an expansion parameter for the width of embeddings. We do not include a comparison with a directed LVN in the main evaluation, as it does not align with the mathematical framework of LEED. However, we conducted additional experiments to explore this setting, which can be found in VII-A, we leave further exploration of this possibility for future work. Whenever the code was available, we ran the experiments ourselves. For all our experiments, we use a server with 2 GPU NVIDIA RTX 6000 Ada Generation (49 Go).

## VII. RESULTS

Table I summarizes our results. It is clear that LVN with LEED performs globally better than any other model. We also obtain better results with LVN-LEED on almost all datasets compared to classical LVN. In addition, PANDA-LEED achieves strong performance on the first three datasets. For graph-structured datasets with informative node embeddings, such as MUTAG, ENZYMES, and PROTEINS, the results indicate that LEED reliably identifies critical nodes, which can subsequently be leveraged to construct meaningful virtual nodes or enlarged node representations. In contrast, for datasets without intrinsic node embeddings, LEED also attains competitive performance, yielding good results in most cases. The only exception is COLLAB within the PANDA framework. However as PANDA alone performs even worse than classical GCN on this dataset, it is not surprising to

TABLE I  
CLASSIFICATION ACCURACY (%) ON BENCHMARK DATASETS, WITH **FIRST**, **SECOND** AND **THIRD** BEST SCORES

| Model       | MUTAG               | ENZYMES             | PROTEINS            | REDDIT-Binary       | IMDB-Binary         | COLLAB              | Avg. Rank |
|-------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|-----------|
| GCN         | 74.750±4.030        | 29.083±2.363        | 71.410±1.077        | 77.735±1.586        | 60.500±2.729        | <b>70.490±1.628</b> | 5.1       |
| Cayley [26] | <b>83.750±3.597</b> | 31.000±2.397        | 73.036±1.291        | 67.050±1.483        | 56.200±1.825        | 69.630±0.730        | 4.8       |
| LVN [29]    | 82.333±2.149        | <b>31.367±1.376</b> | 73.189±0.882        | <b>83.440±0.775</b> | <b>66.620±1.513</b> | <b>71.520±0.661</b> | 3         |
| PANDA [30]  | <b>86.068±2.198</b> | <b>31.550±1.230</b> | <b>75.647±1.076</b> | <b>80.690±0.721</b> | 63.760±1.012        | 68.433±1.012        | 3         |
| PANDA-LEED  | <b>86.838±1.653</b> | <b>33.182±1.921</b> | <b>75.781±1.413</b> | 80.488±0.946        | <b>64.700±1.748</b> | 67.785±1.001        | 2.6       |
| LVN-LEED    | 83.333±2.325        | 31.094±1.748        | <b>74.775±1.066</b> | <b>84.330±0.979</b> | <b>68.060±1.533</b> | <b>72.684±0.567</b> | 2.3       |

observe a performance degradation, even if not statistically significant, when attempting to further refine PANDA. These findings suggest that even in the absence of node features, LEED provides a reliable estimator of node importance, performing comparably to, and in many cases better than, existing approaches. Finally, it is worth mentioning that training increasingly large models with numerous hyperparameters becomes progressively more expensive. Hence, achieving better or even comparable performance with fewer parameters is a highly encouraging result. Using only LEED instead of testing several classical distance metrics to find critical nodes is definitely more efficient.

#### A. Supplementary tests

In Table II we show classical metrics and Dirichlet energy complexity for an unweighted sparse graph to compare with LEED and in particular  $T_2$  function. With  $m = |E|$ ,  $n = |V|$ ,  $l$  the dimension of embeddings and  $\bar{d}$  the mean degree of nodes. It shows that for sparse graphs, where  $\bar{d}$  is low, LEED will be approximately as fast as Dirichlet energy.

TABLE II  
COMPLEXITY CENTRALITY METRICS VS LEED

|                   |                            |
|-------------------|----------------------------|
| PageRank (k iter) | $\mathcal{O}(k.m)$         |
| Closeness         | $\mathcal{O}(n.m)$         |
| Betweenness       | $\mathcal{O}(n.m)$         |
| Dirichlet         | $\mathcal{O}(l.(m+n))$     |
| LEED              | $\mathcal{O}(l.n.\bar{d})$ |

The framework LVN is also producing a directed graph GCN model. Since we developed LEED with a symmetric adjacency matrix in mind, and most of the literature and available mathematical tools assume this structure, we did not investigate this aspect in depth. However, we conducted several tests with directed LVN for comparison, using its original values. As shown in Table III, we obtain results comparable to prior work on MUTAG and PROTEINS, while significantly improving the previous best performance on ENZYMES.

#### B. Critical nodes on over-smoothing

As previously discussed, over-smoothing is no longer universally regarded as either beneficial or detrimental; its impact is highly task-dependent, and different forms or degrees

TABLE III  
DIRECTED LVN COMPARISON  
\* BEST AMONG: {DEGREE, BETWEENNESS, CLOSENESS, PAGERANK}

|                   | MUTAG               | ENZYMES             | PROTEINS            |
|-------------------|---------------------|---------------------|---------------------|
| original metrics* | <b>84.778±2.902</b> | 31.400±2.095        | 74.360±1.105        |
| with LEED         | 83.444±2.694        | <b>37.773±1.669</b> | <b>74.793±1.138</b> |

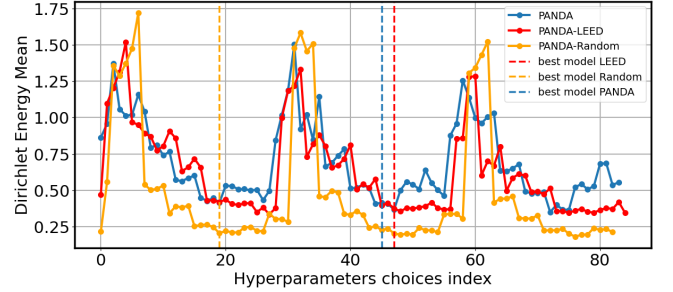


Fig. 7. Dirichlet energy over sets of hyperparameters for LEED and other metrics in MUTAG dataset

of over-smoothing can either degrade or improve performance [21]. Architectural modifications such as introducing virtual nodes or increasing the embedding width of selected nodes typically exacerbate over-smoothing. While prior works has largely focused on mitigating this effect, over-smoothing can also be leveraged to facilitate the propagation of important information and, in some cases, improve the model performance. To assess the impact of critical node selection on over-smoothing, we compare PANDA with LEED and random selection across varying hyperparameters. Figure 7 illustrates that optimal hyperparameter configurations are not necessarily associated with a reduction of over-smoothing. We observe that randomly selecting central nodes in the PANDA framework tends to increase the risk of over-smoothing, whereas our critical node selection achieves performance comparable to that obtained with the best-performing classical centrality measures.

## VIII. CONCLUSION

In this paper, we introduced LEED, a local metric for quantifying over-smoothing that preserves a global behavior comparable to Dirichlet energy while enabling node-level analysis. We demonstrated that this locality can be effectively exploited within virtual and expanded nodes mechanisms

to mitigate over-squashing. Our results suggest that over-smoothing and over-squashing should be addressed jointly, as they are intrinsically linked through the dynamics of message-passing in GNNs and the graph topology.

Empirically, LEED outperforms most classical centrality measures when used to guide both PANDA and LVN construction across a wide range of graph-level tasks, indicating strong consistency in the LEED metric. Our experimental evaluation is restricted to the GCN architecture, whereas many rewiring and virtual node methods have also been explored in conjunction with alternative architectures such as GIN. Extending the analysis to these models constitutes a natural direction for future work. Moreover, while we focus on the application of LEED within virtual node frameworks, the broader implications of local over-smoothing analysis remain largely unexplored. Leveraging LEED to design novel, adaptive graph rewiring strategies represents a promising avenue for further investigation.

## REFERENCES

- [1] A. Paul, S.-V. José, L. Andra, C.-A. Albert, and B.-R. Pere, “Enhancing 5G radio planning with graph representations and deep learning,” in *5G-MeMU* ’23, 2023.
- [2] B. Rémi and Z. Tahar, “Fault detection in telecom networks using bi-level federated graph neural networks,” in *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2023.
- [3] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
- [4] K. Maciej, Łukasz Szymon, and G. A. H., “Graph neural networks in computer vision - architectures, datasets and common approaches,” in *International Joint Conference on Neural Networks (IJCNN)*, 2022.
- [5] C. Kaixuan, L. Shunyu, Z. Tongtian, Q. Ji, S. Yun, T. Yingjie, Z. Tongya, Z. Haoifei, F. Zunlei, Y. Jingwen, and S. Mingli, “Improving expressivity of GNNs with subgraph-specific factor embedded normalization,” in *ACM SIGKDD* ’23, 2023.
- [6] R. T. Konstantin, B. M. M., and M. Siddhartha, “A survey on over-smoothing in graph neural networks,” *arXiv preprint arXiv:2303.10993*, 2023.
- [7] S. Akansha, “Over-squashing in graph neural networks: A comprehensive survey,” *Neurocomputing*, vol. 642, p. 130389, 2025.
- [8] W. Felix, S. Amauri, Z. Tianyi, F. Christopher, Y. Tao, and W. Kilian, “Simplifying graph convolutional networks,” in *ICML*’19, 2019.
- [9] K. Oono and T. Suzuki, “Graph neural networks exponentially lose expressive power for node classification,” *arXiv preprint arXiv:1905.10947*, 2019.
- [10] C. Tianlong, Z. Kaixiong, D. Keyu, Z. Wenqing, W. Peihao, H. Xia, and W. Zhangyang, “Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 2769–2781, 2023.
- [11] J. Zhimeng, L. Zirui, H. Xiaotian, Q. Feng, J. Hongye, T. Qiaoyu, Z. Kaixiong, Z. Na, and H. Xia, “Gradient rewiring for editable graph neural network training,” in *NeurIPS*, 2024.
- [12] Z. Kaixiong, H. Xiao, Z. Daochen, C. Rui, L. Li, C. Soo-Hyun, and H. Xia, “Dirichlet energy constrained learning for deep graph neural networks,” *Advances in neural information processing systems*, vol. 34, pp. 21834–21846, 2021.
- [13] H. Attali, D. Buscaldi, and N. Pernelle, “Rewiring techniques to mitigate oversquashing and oversmoothing in GNNs: A survey,” *arXiv*, 2024.
- [14] P. Liang, H. Rongyao, K. Fei, G. Jiangzhang, M. Yujie, S. Xiaoshuang, and Z. Xiaofeng, “Reverse graph learning for graph neural network,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 4, pp. 4530–4541, 2024.
- [15] D. Wei, Y. Dawei, and W. Peng, “Self-supervised node representation learning via node-to-neighbourhood alignment,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 6, pp. 4218–4233, 2024.
- [16] G. R. Brühl, Y. Mikhail, and S. Justin, “Rewiring with positional encodings for graph neural networks,” *Transactions on Machine Learning Research*, 2023.
- [17] M. Christopher, K. N. M., B. Franka, K. Kristian, M. Petra, and N. Marion, “Tudataset: A collection of benchmark datasets for learning with graphs,” in *ICML*’20, 2020.
- [18] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” in *ICML*’18, 2018.
- [19] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, “measuring and relieving the over-smoothing problem for graph neural networks from the topological view,” in *AAAI*’20, 2020.
- [20] G. J. H., S. Konstantinos, B. Thierry, and M. F. D., “On the trade-off between over-smoothing and over-squashing in deep graph neural networks,” in *ACM International Conference on Information and Knowledge Management (ICKM)*, 2023.
- [21] J. Southern, F. D. Giovanni, M. Bronstein, and J. F. Lutzeyer, “understanding virtual nodes: oversquashing and node heterogeneity,” in *ICLR*’25, 2025.
- [22] C. Chen, H. T. Son, Y. Rose, and W. Yusu, “On the connection between MPNN and graph transformer,” in *ICML*’23, 2023.
- [23] Z. Yuelin, C. Jiacheng, H. Jiaqi, and H. Wenbing, “Fast and distributed equivariant graph neural networks by virtual node learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–15, 2026.
- [24] F. Lukas and W. Melanie, “Mitigating over-smoothing and over-squashing using augmentations of forman-ricci curvature,” in *Learning on Graphs Conference*, 2024.
- [25] T. Jake, D. G. Francesco, C. B. Paul, D. Xiaowen, and B. M. M., “Understanding over-squashing and bottlenecks on graphs via curvature,” *arXiv preprint arXiv:2111.14522*, 2021.
- [26] J. Wilson, M. Bechler-Speicher, and P. Veličković, “Cayley graph propagation,” *ArXiv*, 2025.
- [27] Z. Tan, R. Guo, K. Ding, and H. Liu, “Virtual node tuning for few-shot node classification,” in *ACM SIGKDD* ’23, 2023.
- [28] P. Phu, B. Quang-Thinh, T. N. Ngoc, K. Robert, Y. P. S., and V. Bay, “Topological data analysis in graph neural networks: Surveys and perspectives,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 6, pp. 9758–9776, 2025.
- [29] T. H. Karabulut and İnci M. Baytaş, “Local virtual nodes for alleviating over-squashing in graph neural networks,” *ArXiv*, 2025.
- [30] J. Choi, S. Park, H. Wi, S.-B. Cho, and N. Park, “PANDA: Expanded width-aware message passing beyond rewiring,” in *ICML*’24, 2024.
- [31] Z. Weichen, W. Chenguang, W. Xinyan, H. Congying, G. Tiande, and Y. Tianshu, “Understanding oversmoothing in diffusion-based GNNs from the perspective of operator semigroup theory,” in *ACM SIGKDD* ’25, 2025.
- [32] C. Rongqin, L. Yan, W. Dan, M. Fan, Z. Shenghui, I. P. Lon, I. H. Cheong, L. Ye, and U. L. Hou, “Enhanced subgraph learning in 2-FWL GNNs via local connectivity, spectral, and distance encodings,” in *ACM SIGKDD* ’25, 2025.
- [33] M. B. et al., “Demystifying higher-order graph neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 48, no. 3, pp. 2544–2565, 2026.
- [34] H. Zhu and P. Koniusz, “Simple spectral graph convolution,” in *ICLR*’21, 2021.
- [35] F. Tori, V. Holst, and V. Ginis, “The effectiveness of curvature-based rewiring and the role of hyperparameters in GNNs revisited,” in *ICLR*’25, 2025.