Adaptive Topology Reconstruction for Robust Graph Representation Learning

Anonymous Authors
Under Review

Abstract

Graph Neural Networks (GNNs) have become fundamental in semi-supervised learning for graph representation, leveraging their ability to capture complex node relationships. A recent trend in GNN research focuses on adaptive multi-hop structure learning, moving beyond fixed-hop aggregation to more flexible and dynamic neighborhood selection. While GAMLP [Zhang et al., 2022] employs separate MLP layers for each multi-hop domain and ImprovingTE [Yao et al., 2023] enhances this by injecting contextualized substructure information, these methods still rely heavily on predefined sampling strategies, which may limit their ability to generalize and maintain stable accuracy. To address these limitations, we propose an **adaptive reconstruction framework** that dynamically refines multi-hop structure learning. Inspired by "coreset selection" [Guo et al., 2022], our approach adaptively **reconstructs** node neighborhoods to optimize message passing, ensuring more **effective and context-aware information flow** across the graph. To further enhance structural robustness, we introduce two key modules: the **Distance Recomputator** and the **Topology Reconstructor** (DRTR). The Distance Recomputator **reassesses and recalibrates** node distances based on adaptive graph properties, leading to **improved node embeddings** that better reflect latent relationships. Meanwhile, the Topology Reconstructor **dynamically refines local graph structures**, enabling the model to **adapt to evolving graph topologies** and mitigate the impact of noise and mislabeled data. Empirical evaluations demonstrate that our **adaptive reconstruction framework** achieves **significant improvements** over existing multi-hop-based models, providing more **stable and accurate** performance in various graph learning benchmarks.

1 Introduction

Existing GNN architectures predominantly adopt **static node representations** and **predefined aggregation schemes**, which can be suboptimal when handling graphs with dynamic topologies. Furthermore, the computation of node distances in large-scale graphs introduces significant overhead, making it challenging to scale these models effectively [Kung et al., 2024, Wang et al., 2024]. These limitations highlight the need for a more flexible and efficient approach that can dynamically **reconstruct** graph structures, allowing models to better adapt to changes in connectivity patterns and improve representation quality.

To address these challenges, we propose an **adaptive reconstruction framework** that refines both node distance calculations and local topological structures to enhance message passing efficiency. This framework introduces two key components: the **Distance Recomputator** and the **Topology Reconstructor** (DRTR). The Distance Recomputator dynamically **recalibrates node distances** within the multi-hop domain by incorporating an adaptive encoding mechanism that adjusts to changes in graph density and node proximity. By refining distance computations, it provides a more precise representation of node relationships, which improves downstream predictions and model stability. The Topology Reconstructor complements this by continuously **adjusting local graph**

connectivity, ensuring that the model remains robust in scenarios where structural variations occur. This adaptive mechanism allows GNNs to dynamically reconstruct neighborhood structures, making them more resilient to graph shifts and variations in node interactions.

Through extensive experiments on multiple benchmark datasets, we demonstrate that our adaptive reconstruction approach significantly improves performance in tasks involving both static and dynamic graphs. The Distance Recomputator proves effective in optimizing node representations, reducing the impact of noisy or misleading edges while enhancing the quality of learned embeddings. Meanwhile, the Topology Reconstructor enables models to better align with evolving graph structures, leading to more stable and accurate predictions. In addition to empirical validation, we provide a **theoretical analysis** that explains the benefits of reconstruction in GNN learning dynamics, offering deeper insights into why adaptively refining graph structures leads to superior model generalization.

2 Motivation

2.1 Topology Imbalance Problem

The structural connections in a graph do not always accurately reflect the significance of each link. As stated in [Liu et al., 2023], "not every link is as useful as its topology suggests." Consider a social network where an account labeled "math" follows ten mathematical educators and one food-maker. For tasks such as label prediction, the connection to the food-maker contributes little relevant information, yet traditional Graph Neural Networks (GNNs) aggregate information from all 1-hop neighbors indiscriminately. This results in a phenomenon we define as the **Topology Imbalance Problem**, where some connections (e.g., links to math educators) carry significantly more useful information than others (e.g., the food-maker link).

While models such as Graph Attention Networks (GAT) assign importance scores to each link, they still rely on **synchronous aggregation**, where all nodes within the same hop are processed simultaneously. This limitation prevents the model from prioritizing more relevant connections over less meaningful ones during message passing. To overcome this issue, we propose the **Distance Recomputator**, which dynamically recalculates node distances to better reflect the contextual importance of each link. Additionally, we introduce the **Asynchronous Aggregator**, which enables nodes to be aggregated based on these recomputed distances, allowing for a more adaptive and information-sensitive update mechanism.

2.2 Limitations of Synchronous Aggregation

Most GNN architectures employ **synchronous aggregators**, where information from a node's immediate neighbors is aggregated simultaneously. For example, Graph Convolutional Networks (GCN) aggregate neighbor information by weighting it with the inverse node degree, while GAT uses an adjacency matrix mask and attention scores to prioritize neighbors. Similarly, GraphSAGE samples a subset of neighbors before applying an aggregation function. Despite their differences in implementation, these models share a common characteristic: all nodes in the same layer are processed concurrently.

A major drawback of synchronous aggregation is its inability to differentiate between links of varying importance in a timely manner. Since all neighboring nodes are updated simultaneously, irrelevant or even erroneous connections can propagate misleading information, reducing model robustness. This issue becomes more pronounced in large and complex graphs, where spurious or noisy edges can significantly degrade performance.

To address this limitation, we propose an **asynchronous aggregation mechanism** that selectively integrates information from different neighbors based on their relevance. Unlike traditional synchronous approaches, our method dynamically determines the order and priority of aggregation, reducing the negative influence of uninformative links while preserving critical structural relationships. Some recent works have explored asynchronous processing in GNNs, such as AEGNN [Schaefer et al., 2022], which updates node activations only upon new events, and Gated Graph Sequence Neural Networks [Li et al., 2017], which employ gated recurrent units to process sequences asynchronously. However, these methods primarily focus on message passing rather than addressing

topological reconstruction at a structural level. Our approach directly enhances the representation of graph topology, offering a more comprehensive solution.

2.3 Heat Diffusion in Graph Neural Networks

The concept of **heat diffusion** offers a valuable perspective for understanding information propagation in Graph Neural Networks (GNNs). Drawing inspiration from thermodynamics, where heat gradually dissipates as it moves away from its source [Thanou et al., 2016], we analyze how information in GNNs follows a similar pattern of attenuation during multi-hop message passing. As information propagates across a graph, its influence weakens with increasing distance from the source node, often leading to a decline in signal quality and an accumulation of noise. Traditional GNN architectures typically assume uniform propagation across all hops, failing to account for this natural decay in information relevance.

To address this limitation, we introduce a **distance-aware reconstruction framework** that models message passing through a heat diffusion-inspired mechanism. Our approach regulates the transmission strength based on a **distance-dependent attenuation function**, ensuring that information decays in a controlled and meaningful manner. By integrating this adaptive reconstruction process, we prevent the excessive spread of weak or irrelevant signals while reinforcing the retention of important structural patterns. This refinement enhances both the **signal-to-noise ratio** and the interpretability of learned node representations, making the model more robust to graph sparsity and noise.

By incorporating heat diffusion principles into the reconstruction process, our method achieves a more balanced and adaptive message propagation strategy. This approach not only improves the efficiency of information flow but also mitigates common issues such as oversmoothing and excessive feature mixing in deep GNN models. Additional theoretical analysis and mathematical formulations of our heat diffusion-based reconstruction mechanism are provided in our supplementary document, "Heat Diffusion in GNNs."

2.4 Paper Contribution

- **K-hop Diffusion Message Passing:** We propose a message passing framework that integrates k-hop information at each propagation step. After each update, the model recalibrates node embeddings and distances to reflect newly acquired information. A heat diffusion-inspired mechanism controls attenuation, preserving meaningful connections while reducing noise.
- **Distance Recomputator:** Our model recalculates node distances using an adaptive attention mechanism that considers both multi-hop topology and feature interactions. It continuously adjusts distances based on contextual relevance, improving message passing and filtering irrelevant edges.
- **Topology Reconstructor:** We introduce a multi-hop topology reconstruction method. our model leverages multi-hop topology from sampling to perform reconstruction based on computed "similarity distances." Nodes exceeding a similarity distance threshold are repositioned to optimize network configurations for enhanced learning outcomes.

3 Core Ideas of DRTR

3.1 K-hop Message Passing

Most existing multi-hop models in graph neural networks rely on multiple rounds of 1-hop message passing to capture higher-order information. However, they lack the ability to perform k-hop message passing with multi-hop propagation in a single iteration or efficiently sample multi-hop neighborhoods in one step. For example, the enhancing multi-hop connectivity model refines multi-hop neighborhoods by sampling multi-hop neighbors and reweighting high-order connections, strengthening highly related nodes while down-weighting weaker ones [Liu et al., 2022]. Similarly, KP-GNN enriches node representations by incorporating peripheral embeddings at each layer [Feng et al., 2022].

Our model introduces an efficient multi-hop sampling strategy that enables direct k-hop message passing, capturing local structures more effectively than traditional 1-hop methods. By integrating multi-hop information in each propagation step, our approach provides a more comprehensive graph perspective, leading to improved label prediction accuracy for central nodes.

3.2 Diffusion Propagation in Graph Neural Networks

We propose a diffusion-based propagation model that builds upon traditional approaches such as the Diffusion Decent Network. One well-known model, MAGNA [Wang et al., 2021], propagates information by updating vertex and edge embeddings while recalculating distances for both one-hop and multi-hop neighbors, considering all paths up to k hops. Another approach, "Diffusion Improves Graph Learning" [Klicpera et al., 2019], approximates the diffusion equation using an infinite series, improving computational efficiency on large-scale graphs.

Our model extends these concepts by simulating heat diffusion within the multi-hop domain. This diffusion-inspired framework regulates message passing, dynamically adjusting the flow of information to reduce noise accumulation and enhance representation quality. By incorporating heat diffusion into multi-hop propagation, our model provides a structured and adaptive approach to information distribution in graph neural networks.

3.3 Graph Imbalance Problems

Label and topology imbalance in graph neural networks remain significant but underexplored challenges. Zhao et al. [Zhao et al., 2022] addressed topology imbalance by adjusting edge weights without modifying the graph structure.

We introduce two models that actively reconstruct graph topology to improve balance. The first, GKHDDRA, applies hop jumping to restructure the graph and refine connectivity. The second, GDRA, enhances the dataset by removing weak edges and forming new connections between strongly similar nodes. These adjustments optimize graph structure, leading to a more balanced and representative network, ultimately improving learning performance.

4 Architectures of DRTR

The Distance Recomputator and Topology Reconstructor (DRTR) is designed to enhance Graph Neural Networks (GNNs) by addressing two major challenges: (1) capturing long-range dependencies while maintaining computational efficiency, and (2) dynamically adjusting the graph topology based on learned representations. The K-hop Diffusion Attention Layer, a core component of DRTR, integrates static preprocessing, multi-hop diffusion, and dynamic topology updates to optimize message passing and representation learning.

Graph-based learning problems often rely on an underlying undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with the node set $\mathcal{V}$ and edge set $\mathcal{E}$. The adjancy matrix A $\in R^{N \times N}$ describes the existence of connections among nodes. Every node $v_i \in \mathcal{V}$ has an associated feature vector $x_i \in R^{d \times 1}$, so the whole feature vector space can be represented as $X = [x_1, x_2, ..., x_N]^T$, so that the goal is to predict the labels of the remaining nodes. For a semi-supervised node classification task on a graph, the labels Y_L are only available for a subset of the noes (training set), so the goal is to predict the labels of the remaining nodes.

In GNN tasks, the topology information and node feature are combined to learn the representation vetor of a node for node tasks. Modern GNNs aggregate the information from nodes in neighborhood and update the representation of the nodes by a message-passing scheme. After k-iterations of aggregation, the representation of a node captures the structural information within its multi-hop neighborhood. Formally, the layer-wise aggregation of a GNN is given by

$$a_v^{(k)} = Aggregate^{(k)}(\{h_u^{k-1} : u \in N(v)\}), h_v^{(k)} = Combine^{(k)}(h_v^{(k-1)}, a_v^{(k)}) \tag{1}$$

where $h_v^{(k)}$ represents the feature vector of node v at the kth layer. The initialization follows $h_v^{(0)} = x_v$, where $\mathcal{N}(v)$ denotes the set of neighboring nodes connected to v.

Existing works primarily focus on improving the 1-hop aggregation $\mathcal{AGGREGATE}^{(k)}(\cdot)$ and combination $\mathcal{COMBINE}^{(k)}(\cdot)$ operations. For example, Graph Convolutional Networks (GCN) utilize a convolution operation defined as:

$$H^{(l+1)} = \sigma(\hat{A}H^{(l)}\Theta^{(l)}) \tag{2}$$

where $H^{(l)}$ represents the feature matrix at the lth layer, initialized as $H^{(0)} = X$. The term $\hat{A} = \hat{D}^{-\frac{1}{2}}(A+I)\hat{D}^{-\frac{1}{2}}$ is the Laplacian-normalized adjacency matrix, and $\Theta^{(l)}$ is the learnable weight matrix at the lth layer.

GNN models stack multiple layers to capture hierarchical representations over a larger neighborhood. Taking a two-layer GCN as an example, the final node representation is computed as:

$$Z = \text{softmax}(\hat{A}\sigma(\hat{A}X\Theta^{(0)})\Theta^{(1)}) \tag{3}$$

For a node classification task, given labeled nodes corresponding to Y_L, the objective function is defined as:

$$L = -\frac{1}{|\mathcal{Y}_\mathcal{L}|}\sum_{v_i \in \mathcal{Y}_\mathcal{L}} y_i log(z_i) \tag{4}$$

4.1 In-Depth Learning in Graph Neural Networks

Contemporary GNNs predominantly employ a 1-hop message passing paradigm, where each propagation step explores information within a single depth level. Unlike GraphSAGE [Hamilton et al., 2017], which samples along a k-depth domain but aggregates at every depth, our approach integrates both sampling and aggregation within a k-hop neighborhood. This enables a more comprehensive understanding of local structural dependencies within the graph.

We propose an in-depth learning algorithm for k-hop sampling and message passing in GNNs, comprising the following key components:

1. Static Preprocessing Step: A large number of neighbors are pre-sampled for each node, constructing a k-hop adjacency matrix that encapsulates extended neighborhood information. This process achieves efficient storage of structural dependencies with a computational complexity of $\mathcal{O}(\mathcal{V}K)$, where $\mathcal{V}$ denotes the total number of vertices.

2. Dynamic Update Phase: During propagation, a refined subset of k-hop neighbors is dynamically sampled, updating the precomputed neighborhood structure. This step ensures that the model adapts to evolving graph topologies while maintaining computational efficiency at $\mathcal{O}(KR)$, where $\mathcal{R}$ represents the number of sampled vertices per propagation step.

The k-hop attentive message passing strategy leverages both static and dynamic sampling pathways. This mechanism is inspired by graph diffusion networks and integrates distance reevaluation indicators derived from the Graph Attention Network (GAT) [Veličković et al., 2018]. By synthesizing static and dynamic sampling methodologies, our approach enhances both the depth and accuracy of message passing in GNNs.

DRTR Diffusion Layer design is an integration of **multi-hop diffusion**, **adaptive message passing**, and **dynamic topology reconstruction** to capture both local and long-range dependencies. Instead of standard **1-hop aggregation**, DRTR employs an **iterative K-hop aggregation** process, where node representations are refined using a combination of **local neighborhood aggregation and global diffusion attention**. To further improve feature propagation, a **Graph Attention Mechanism (GAT)** dynamically computes attention weights, prioritizing important neighbors and filtering out noisy connections. Additionally, DRTR introduces a **Topology Reconstructor Module**, which adjusts edge weights and graph connectivity by **recomputing distances** and **adapting to evolving structures**. This mechanism mitigates over-smoothing and ensures the model remains responsive to structural variations. By balancing **computational efficiency** and **expressive power**, DRTR's **hierarchical multi-hop framework** enables effective large-scale graph learning while maintaining a trade-off between depth (global context) and locality (fine-grained details).

Algorithm 1: Static Sampling and Dynamic Resampling

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Depth K, Sampling number per hop N,
Input features $\mathbf{Z}$, Adjacency Matrix $\mathbf{A}$
Output: $\mathcal{NGH}$: K-hop Sampling Storage (3D dictionary)

```
NGH^(0) ← A                                   // Initialize neighborhood
// Static Sampling before Computation
for n ∈ V do
    for k = K ... 1 do
        SN = RS(N, Layer^(k−1))
        for v_i ∈ Layer(n, k − 1) do
            NGH_n^(k) ← NGH_n^(k) ∪ GraphSAGE(v_i, SN[i])
        end
    end
end
// Dynamical Re-Sampling during Computation
B^(K) ← B // B:  Batch of nodes to process
for k = K ... 1 do
    B^(k−1) ← B^(k)
    for u_i ∈ B do
        SN_s ← RS(u_i, N)
        NGH_n^(k) ← NGH_n^(k) ∪ GraphSAGE(u_i, SN_s[i])
        z_{u_i} ← GKHDA(u_i, k, NGH_n^(k))
    end
end
```

In our Graph Neural Network framework, the implementation of the $\mathcal{COMBINE}$ function plays a pivotal role. It can be expressed mathematically as:

$$z_u \leftarrow W_i \cdot h_u^i, \quad \forall i \in \text{range}(1, K+1) \tag{5}$$

where z_u represents the combined feature vector for a node u, and h_u^i denotes the feature vector of the node at the i-th hop. W_i is the weight matrix corresponding to the i-th hop, ensuring that features from different hops are weighted differently in the aggregation process.

Similarly, the $\mathcal{GAT}$ function, pivotal in our model for attention mechanism, is formulated as:

$$\alpha_{ij} = \frac{\text{LeakyReLU}(\vec{a}^T[W\hat{h_i} \,||\, W\hat{h_j}])}{\sum_{k \in N_i} \text{LeakyReLU}(\vec{a}^T[W\hat{h_i} \,||\, W\hat{h_k}])} \tag{6}$$

Here, α_{ij} is the attention coefficient between nodes i and j, calculated using the LeakyReLU activation function. $\vec{a}$ is the attention vector and W is the weight matrix applied to the feature vectors $\hat{h_i}$ and $\hat{h_j}$ of the nodes i and j. The attention mechanism effectively captures the importance of each neighbor's features in the aggregation process.

In this paper, we adopt a structure that considers both k-hop sampling and k-hop message passing within the graph diffusion network framework. Striking a balance between computational efficiency (time) and model performance, our approach involves static preprocessing for initial multi-hop neighborhood sampling, followed by dynamic resampling in subsequent iterations. This methodology ensures that our model remains efficient while effectively capturing the complex dependencies in the graph structure across multiple hops.

4.2 Distance Recomputator and Topology Reconstructor Implementation

Distance Recomputator and Topology Reconstructor is meticulously designed to dynamically reconfigure graph topology and recalibrate node distances, thereby significantly enhancing the representational capacity and performance of GNNs.

Initial Setup and K-Hop Sampling: The algorithm begins with an initial setup phase where it prepares the graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with input features x_v for all vertices v in the batch $\mathcal{B}$. Each node is

Algorithm 2: K-hop Diffusion Attention Layer Implementation

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Depth K, Non-linearity σ,
Input features $\mathbf{x_v}$, Weight matrices $\mathbf{W^k}$,
Aggregator functions $\mathcal{AGGREGATE}_k, \forall k \in \{1, ..., K\}$,
Neighborhood sampling $N_k : v \rightarrow 2^v, \forall k \in \{1, ..., K\}$,
$\mathcal{NGH} \leftarrow$ StaticSampling$(\mathcal{G}, K, \mathbf{Z}, \mathbf{A})$

Output: Vector Representation $\mathbf{z_v}$ for all $v \in \mathcal{B}$

```
// Static Aggregation
```

$h_u^0 \leftarrow x_v, \forall v \in \mathcal{B}$

for $k = 1 \ldots K$ **do**
  for $u \in \mathcal{B}^{(k)}$ **do**
    $h_u^k \leftarrow \sigma(\mathbf{W}^k \cdot (h_u^{k-1}, h_{N(u)}^k))$
    $h_u^k \leftarrow \frac{h_u^k}{\|h_u^k\|_2}$
  end
end

```
// K-hop Diffusion Attention
```

for $k = K \ldots 1$ **do**
  for $u \in \mathcal{NGH}_u^{(k)}$ **do**
    $h_{N(u)}^k \leftarrow \mathcal{AGGREGATE}_k\{\mathcal{GAT}(h_{u'}^{k-1}, \mathbf{W}^k) \mid u' \in N_k(u)\}$
  end
end

$\mathbf{z_u} \leftarrow \mathcal{COMBINE}(h_u^i), \forall i \in \{1, \ldots, K\}$

associated with a depth K, represented by weight matrices W^k, and a non-linearity function σ. The model incorporates differentiable aggregator functions $\mathcal{AGGREGATE}_k$ for each depth k, along with neighborhood sampling functions N_k. These components collectively form the foundation for k-hop neighborhood sampling, a crucial step in our model's operation.

Dynamic Resampling and Recomputation: At the heart of our model lies a dynamic resampling process, which is executed in each propagation phase. This process adapts to the evolving graph structure by resampling a smaller subset of k-hop neighbors for each node. The resampled data, represented by NGH (Neighborhood Graph), is then refined through a series of computations to update the node representations effectively.

Distance Recomputator Mechanism: Central to our model is the Distance Recomputator (DRM), which recalculates the distances between nodes based on certain criteria, including recompute distance bounds α and β, and a sampling factor γ. This mechanism is crucial for adjusting the topological structure of the graph, ensuring that nodes are positioned optimally based on their relational context within the graph.

Graph Attention Network (GAT) Integration: The model integrates the Graph Attention Network (GAT) to compute attention coefficients between nodes. This integration allows the model to weigh the importance of each neighbor's features in the aggregation process, further refining the distance recomputation and neighborhood sampling.

Asynchronous Aggregation: Another key aspect of our model is the implementation of an asynchronous aggregation approach. Unlike traditional methods that aggregate information simultaneously across all nodes, our model allows for selective and time-staggered aggregation based on the dynamic resampling and recomputation results. This approach ensures a more nuanced and efficient processing of graph data, crucial for handling large-scale and complex networks.

In summary, our algorithm presents a novel framework that combines multi-hop sampling with dynamic resampling and recomputation, all underpinned by an asynchronous aggregation strategy. This comprehensive approach addresses key challenges in GNNs, such as handling complex graph topologies and efficiently updating node representations, thereby setting a new standard in the field of graph neural network research.

Algorithm 3: Distance Recomputator and Topology Reconstructor

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Depth K, Non-linearity σ,
Input features $\mathbf{x_v}$, Weight matrices $\mathbf{W^k}$,
Aggregator functions $\mathcal{AGGREGATE}_k, \forall k \in \{1, ..., K\}$,
Neighborhood sampling $N_k : v \rightarrow 2^v, \forall k \in \{1, ..., K\}$,
Distance bounds α, β, Sampling factor γ,
$\mathcal{NGH} \leftarrow [0]_{V \times (K \times V)}$, $\mathbf{W} \leftarrow [\delta_{ij}]_{K \times F}$

Output: Vector Representation $\mathbf{z_v}$ for all $v \in \mathcal{B}$

```
// K-hop Sampling
```

$\mathcal{NGH} \leftarrow$ K-Hop-Sample($\mathcal{G}$)

for *Each Propagation* **do**

  $\mathcal{NGH} \leftarrow \mathcal{NGH} \cup$ Resample($\mathcal{G}$, ResampleNum)

  $\mathcal{NGH}_{\text{compute}} \leftarrow$ Sample($\mathcal{NGH}, \gamma$)

  $\mathbf{Z_v} \leftarrow$ GKHDDRA($\mathcal{G}, \mathcal{NGH}_{\text{compute}}$)

  $\mathbf{DRM} \leftarrow$ mask($\mathcal{GAT}(\mathcal{G}, \mathbf{Z_v}), \mathcal{NGH}$)

  `// Distance Recomputing`

  for $i = K - 1 \ldots 0$ **do**

    $\mathbf{DRM}[i+1] \leftarrow \mathbf{DRM}[i+1] + \mathbf{DRM}[i][\mathbf{DRM}[i] > \alpha]$

    $\mathbf{DRM}[i] \leftarrow \mathbf{DRM}[i] - \mathbf{DRM}[i][\mathbf{DRM}[i] > \alpha]$

  end

  `// Topology Reconstruction`

  for $i = 1 \ldots K$ **do**

    $\mathbf{DRM}[i] \leftarrow \mathbf{DRM}[i] + \mathbf{DRM}[i-1][\mathbf{DRM}[i-1] < \beta]$

    $\mathbf{DRM}[i-1] \leftarrow \mathbf{DRM}[i-1] - \mathbf{DRM}[i-1][\mathbf{DRM}[i-1] < \beta]$

  end

end

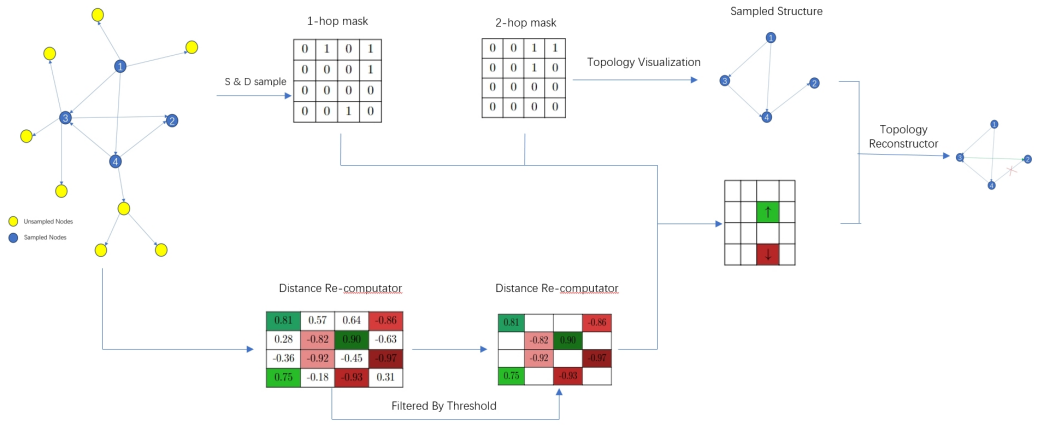

Figure 1: An instance of The Distance Computation and Topology Reconstruction

4.3 Experimental Analysis

Our experimental study focused on evaluating the performance of the proposed models - GKHDA, GDRA, and GKHDDRA - alongside typical scalable Graph Neural Network (GNN) methods, including GCN, SGC, S2GC, and APPNP. The experiments concentrated on graph structure learning and were conducted on benchmark datasets: Cora, Pubmed, and Citeseer.

Performance Overview: The results, as presented in Table 1, showcase the efficacy of our models in comparison with established GNN methods like GAT, GraphSAGE, GCN, SGC, SSGC, and APPNP. Notably, our models - particularly when integrated with existing GNN structures (GCN, SGC, SSGC, and APPNP) - demonstrate superior performance across all datasets.

Dataset-Specific Analysis:

- *Cora*: In the Cora dataset, the highest performance was observed with the APPNP+GKHDDRA combination, achieving an accuracy of 84.6%. This result indicates the robustness of GKHDDRA when combined with APPNP's propagation scheme, which effectively leverages long-range dependencies in the graph.

- *Pubmed*: The SGC+GKHDDRA combination outperformed other models with an accuracy of 82.5%. This suggests that the structured sparsity imposed by SGC, coupled with the advanced hop-wise learning capability of GKHDDRA, is particularly effective for the Pubmed dataset's topology.

- *Citeseer*: For Citeseer, the SSGC+GKHDDRA combination achieved the highest accuracy at 75.6%. This underscores the effectiveness of incorporating multi-hop sampling and dynamic resampling in dealing with the dataset's complex graph structure.

Comparative Assessment: The integration of our models with existing GNN frameworks consistently improved performance across datasets. For instance, GCN, when enhanced with GDRA and GKHDDRA, saw notable improvements in accuracy, emphasizing the value added by our distance recomputation and dynamic resampling mechanisms. Similarly, the integration with SGC and SSGC yielded significant performance boosts, highlighting the synergy between our models and scalable GNN methods in handling large-scale graph structures.

Model-Specific Contributions:

- *GKHDA* showcased consistent improvements in graph representation learning, particularly in combination with SSGC and APPNP.

- *GDRA* excelled in recalibrating the graph topology, which was evident from its strong performance, especially when combined with SGC and SSGC.

- *GKHDDRA* emerged as a versatile model, enhancing both graph structure learning and node representation, as reflected in its top-tier results across all datasets, particularly when combined with APPNP.

Conclusion: The experimental results validate the effectiveness of our proposed models in enhancing the learning capabilities of GNNs. The integration of GKHDA, GDRA, and GKHDDRA with existing scalable GNN methods not only improved performance but also demonstrated their adaptability and compatibility with different graph structures and datasets. These findings indicate that our models are not only theoretically sound but also practically potent in a variety of real-world graph learning scenarios.

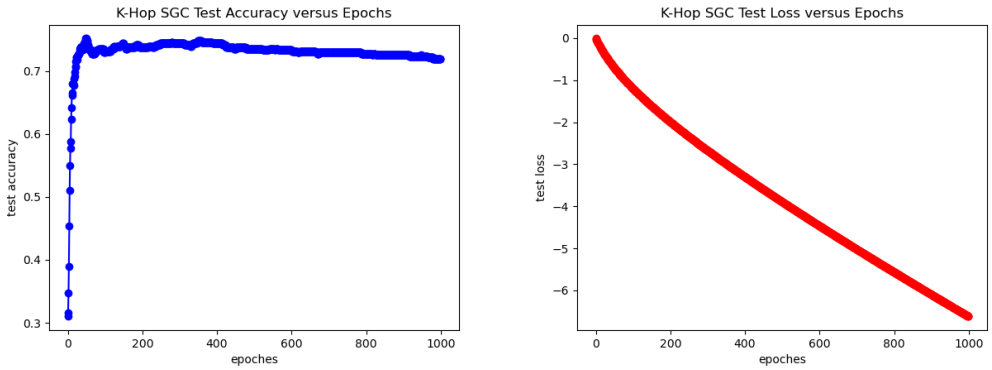

Figure 2: K-Hop SGC (Max Test Acc: 74.2%)

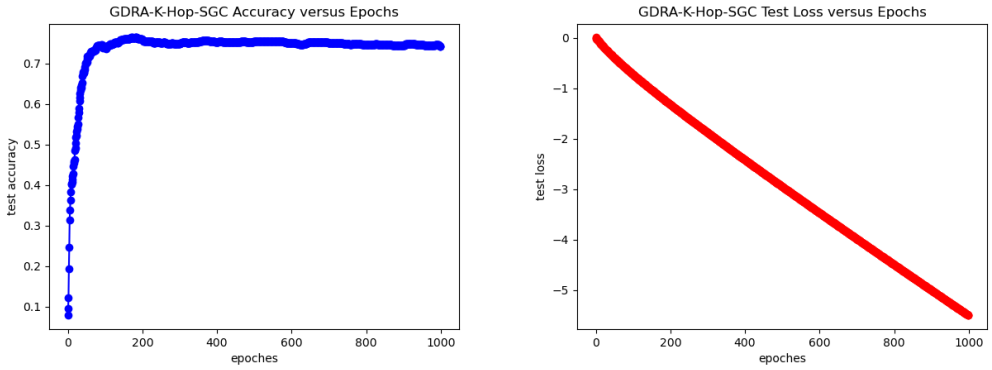

Figure 3: GDRA + K-Hop SGC (Max Test Acc: 75.8%)

References

Jiarui Feng, Yixin Chen, Fuhai Li, Anindya Sarkar, and Muhan Zhang. How powerful are k-hop message passing graph neural networks. 05 2022. doi: 10.48550/arXiv.2205.13328.

Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning, 2022.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017. URL http://arxiv.org/abs/1706.02216.

Johannes Klicpera, Stefan Weißenberger, and Stephan Günnemann. Diffusion improves graph learning. 2019. URL https://api.semanticscholar.org/CorpusID:202783932.

Pau Perng-Hwa Kung, Zihao Fan, Tong Zhao, Yozen Liu, Zhixin Lai, Jiahui Shi, Yan Wu, Jun Yu, Neil Shah, and Ganesh Venkataraman. Improving embedding-based retrieval in friend recommendation with ann query expansion. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2930–2934, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704314. doi: 10.1145/3626772.3661367. URL https://doi.org/10.1145/3626772.3661367.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks, 2017.

Songtao Liu, Shixiong Jing, Tong Zhao, Zengfeng Huang, and Dinghao Wu. Enhancing multi-hop connectivity for graph convolutional networks. 2022. URL https://openreview.net/forum?id=b-fUjY16Y_.

Zemin Liu, Yuan Li, Nan Chen, Qian Wang, Bryan Hooi, and Bingsheng He. A survey of imbalanced learning on graphs: Problems, techniques, and future directions, 2023.

S. Schaefer, D. Gehrig, and D. Scaramuzza. Aegnn: Asynchronous event-based graph neural networks. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12361–12371, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.01205. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01205.

Dorina Thanou, Xiaowen Dong, Daniel Kressner, and Pascal Frossard. Learning heat diffusion graphs, 2016.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. Multi-hop attention graph neural network. 2021.

Zeyu Wang, Yue Zhu, Zichao Li, Zhuoyue Wang, Hao Qin, and Xinqi Liu. Graph neural network recommendation system for football formation. *Applied Science and Biotechnology Journal for Advanced Research*, 3(3):33–39, 2024.

Tianjun Yao, Yingxu Wang, Kun Zhang, and Shangsong Liang. Improving the expressiveness of k-hop message-passing gnns by injecting contextualized substructure information. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023. URL https://api.semanticscholar.org/CorpusID:260499770.

Wentao Zhang, Ziqi Yin, Zeang Sheng, Yang Li, Wen Ouyang, Xiaosen Li, Yangyu Tao, Zhi Yang, and Bin Cui. Graph attention multi-layer perceptron. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22. ACM, August 2022. doi: 10.1145/3534678.3539121. URL http://dx.doi.org/10.1145/3534678.3539121.

Tianxiang Zhao, Dongsheng Luo, Xiang Zhang, and Suhang Wang. Topoimb: Toward topology-level imbalance in learning from graphs. 2022. URL https://openreview.net/forum?id=nR3rZ4ODtQ.

A Appendix

	Cora	Pubmed	Citeseer
GAT	82.1%	77.7%	69%
GraphSAGE	81.5%	70.3%	74%
GDRA	82.5%	77.8%	69.8%

Table 1: Comparision of GDRA and its basic two components (GAT and GraphSAGE)

	Cora	Pubmed	Citeseer
GKHDA	82.3%	78.6%	70.7%
GDRA	82.5%	77.8%	69.8%
GKHDDRA	82.4%	79.4%	71.2%

Table 2: Comparison of 3 Basic Model of DRTR

	Cora	Pubmed	Citeseer
GCN	81.2%	79.3%	70.9%
GCN+GDRA	82.6%	80.1%	71.3%
GCN+GKHDA	82.4%	80.5%	71.7%
GCN+GKHDDRA	82.7%	80.9%	72.3%

Table 3: GCN and SGC + DRTR + Diff comparison

	Cora	Pubmed	Citeseer
SGC	74.2%	78.2%	71.5%
SGC+GDRA	75.8%	81.2%	73.1%
SGC+GKHDA	75.1%	81.6%	73.4%
SGC+GKHDDRA	77.4%	82.5%	74.6%

Table 4: SGC and SGC + DRTR + Diff comparison

	Cora	Pubmed	Citeseer
SSGC	83.0%	73.6%	75.6%
SSGC+GDRA	83.2%	74.2%	76.4%
SSGC+GKHDA	84.3%	74.5%	76.1%
SSGC+GKHDDRA	84.1%	74.7%	77.6%

Table 5: SSGC and SSGC + DRTR + Diff comparison

	Cora	Pubmed	Citeseer
APPNNP	82.3%	71.5%	75.2%
APPNP+GDRA	83.5%	73.6%	74.4%
APPNP+GKHDA	83.8%	74.1%	74.5%
APPNP+GKHDDRA	84.6%	74.5%	75.3%

Table 6: APPNP and APPNP + DRTR + Diff comparison

Dataset	**Nodes**	**Edges**	**Features**	**Classes**
PubMed	19,717	44,338	500	3
Cora	2,708	5,429	1,433	7
CiteSeer	3,312	4,732	3,703	6

Table 7: Comparison of PubMed, Cora, and CiteSeer in Terms of Nodes, Edges, Features, and Classes

Models	DR	TR	k_hop_resampling	Heat_Diffusion_Propagation
GDRA	✓	✓		
GKHDA			✓	✓
GKHDDRA	✓	✓	✓	✓

Table 8: GDRA, GKHDRA, GKHDDRA Modules (DR: Distance Recomputator; TR: Topology Reconstructor)

Table 9: Experimental Settings

Learning Rate	Weight Decay	Epochs	Patience
0.005	0.001	1000	100