GNN Efficiency Improvements by Using Graph Sampling Strategies

Kunxiao Gao

November 2024

1 Introduction

Graph Neural Networks (GNNs) have emerged as a powerful paradigm for processing data with graph structures, excelling in tasks ranging from social network analysis to recommendation systems and molecular property prediction [1, 2]. By leveraging graph convolutions, GNNs efficiently aggregate information from local neighborhoods, enabling superior performance in representing and predicting complex relationships in networked data. However, as the size of the graph grows, the computational cost and memory requirements of GNNs become significant bottlenecks, limiting their applicability in real-time and large-scale systems [3, 4].

A critical challenge lies in constructing and processing graph representations that maintain the predictive power of the full graph while reducing its computational complexity. Traditional GNN implementations often consider the full graph or dense subgraphs, which involve a high number of edges and nodes, leading to inefficiencies. This motivates the exploration of graph signal sampling, a strategy that selects subsets of nodes to construct reduced representations of the graph. These sampled subgraphs aim to preserve the essential characteristics of the original graph, enabling accurate predictions while achieving significant reductions in computational overhead [5].

This study investigates the impact of graph sampling strategies on GNN efficiency. Building upon the theoretical groundwork of Ruiz et al. [5], who analyzed the transferability and scalability of GNNs across graph families defined by graphons, this work focuses on practical sampling techniques that optimize computational performance without compromising accuracy. Specifically, we explore sampling methods on a movie rating network, hypothesizing that selective node inclusion can achieve comparable or superior predictive performance to full graph approaches, with markedly lower resource demands.

By combining theoretical insights with empirical validation, this paper aims to identify and validate optimal sampling strategies, contributing to the broader goal of making GNNs more scalable and efficient. This research not only advances the state of GNN optimization but also has practical implications for applications requiring real-time processing of large-scale graphs, such as social media analytics and dynamic network monitoring.

2 Related Work

2.1 Graphon Neural Networks

The concept of Graphon Neural Networks (GNNs), introduced by Ruiz et al. [6], bridges the gap between finite and infinite graphs by leveraging the mathematical construct of graphons. A graphon is essentially the limit object of a converging sequence of graphs, allowing researchers to study GNN behavior at scale. This framework has proven invaluable in analyzing how GNNs behave when the graph size grows indefinitely while retaining similar structural patterns. It provides a robust way to model large-scale graphs theoretically and computationally.

Graphon Neural Networks offer insights into scalability and convergence, addressing a key question: How can GNNs be generalized to represent larger or infinite graphs while preserving critical properties? In our context, the graphon perspective aligns with our goal of efficient graph sampling. By prioritizing nodes with high leverage scores—essentially nodes that contribute most to the graph's structural and spectral properties—we ensure that the sampled graph retains the "essence" of the original. This approach draws directly from the principles of graphon theory, enabling scalability while preserving representativeness.

Additionally, the graphon framework allows us to connect our sampling techniques with rigorous error bounds, ensuring that the reduced graph does not diverge significantly from the original in terms of predictive performance. By integrating the structural insights provided by graphon theory, our work contributes to bridging theoretical understanding and practical application in graph sampling.

2.2 Transferability

The transferability of GNNs—how well models trained on one graph can generalize to another—has been a longstanding challenge in the field. Ruiz et al. [7] tackled this issue comprehensively, providing a theoretical foundation to understand when and why GNNs can generalize effectively across graphs. Their findings emphasized the role of spectral properties, particularly the similarity of eigenvalues in the normalized Laplacian matrices of different graphs, in ensuring transferability. This spectral perspective is critical for understanding not just generalization, but also the robustness of GNNs across varying graph sizes and structures.

In our work, transferability is addressed through the lens of graph sampling. By sampling nodes using leverage scores—calculated based on the covariance matrix—we ensure that the sampled graph retains critical spectral properties of the original. This approach builds directly on the findings of Ruiz et al. [7], which highlight the importance of preserving spectral alignment for effective GNN transferability. Our method ensures that even with a reduced graph, the model's learned representations remain robust and transferable, making it possible to generalize to new data while maintaining computational efficiency.

2.3 Sampling Strategies

Sampling strategies have long been studied as a means to reduce graph size while preserving critical structural properties. Two common approaches are random sampling and breadth-first search (BFS)-based sampling, each with its advantages and limitations.

Random Sampling is straightforward, where nodes or edges are sampled uniformly at random. This method is computationally efficient but often fails to account for node importance, potentially discarding nodes that are critical to the graph's structure. Studies such as Leskovec and Faloutsos [8] have demonstrated that random sampling can result in reduced representativeness, especially in graphs with highly heterogeneous node degrees.

Breadth-First Search (BFS)-Based Sampling offers an alternative by retaining local connectivity through traversal. While this approach is effective for preserving local graph structures, it introduces biases, particularly toward highly connected nodes, as shown in Ahmed et al. [9]. This bias can lead to the exclusion of globally important nodes that may not be well-connected locally, limiting the generalizability of downstream tasks.

In contrast, leverage score sampling provides a principled framework that directly accounts for node importance. Leverage scores, rooted in linear algebra, quantify how much each node contributes to the graph's overall structure, offering a mathematically grounded way to prioritize nodes for sampling. As highlighted in Mahoney and Drineas [10], leverage scores are particularly useful in preserving spectral properties, making them well-suited for applications involving GNNs, where the spectral domain plays a crucial role in information propagation.

2.4 Limitations of Previous Studies

While previous work has significantly advanced our understanding of graph sampling and GNN scalability, several limitations remain that motivate our approach:

Transferability Across Heterogeneous Graphs: The transferability of GNNs depends on spectral similarity between graphs. While this is effective for structurally similar graphs, it is less applicable to scenarios where graphs have highly diverse eigenvalue distributions, such as molecular graphs versus recommendation networks [7]. Ensuring transferability across such heterogeneous graphs requires methods that preserve both global and local structural properties.

Challenges in Existing Sampling Methods: Traditional sampling approaches, such as random node sampling and BFS-based sampling, have notable drawbacks. Random sampling, though simple, often discards structurally important nodes, leading to significant information loss [8]. BFS-based sampling, while better at preserving local connectivity, introduces bias toward highly connected nodes, which may not reflect global graph properties [9]. These limitations highlight the need for more principled sampling strategies that balance computational efficiency with structural preservation.

3 Preliminaries Definitions about GNNs Structure

Graph Neural Networks (GNNs) are a class of machine learning models that operate directly on graph-structured data, leveraging the graph's topology and node features to perform tasks like node classification, link prediction, and graph-level predictions. Unlike traditional neural networks, which process data in Euclidean spaces, GNNs are specifically designed to exploit the relational and structural properties of non-Euclidean data, such as social networks, molecular structures, and recommendation systems.

Graphs and the Graph Shift Operator (GSO) To understand GNNs, it is essential to recognize the role of the Graph Shift Operator (GSO). A graph G = (V,E,S) consists of:

- **V = Nodes**: Representing individual entities, such as users or molecules.;
- **E = Edges**: Capturing relationships or interactions between nodes.
- **S = Edge Weights**: A matrix representation of the graph topology, with examples including the adjacency matrix A or graph Laplacian L.

The GSO S defines how signals $x \in R^n$ propagate across the graph. For any node i, the operation $x = Sx$, aggregates signal values from its neighboring nodes j, weighted by the proximity measure $[S]_{ij}$ This aggregation forms the foundation of GNNs, where learning relies on propagating information across the graph while respecting its structure [2, 6, 7].

The Power of GNNs Building on this foundation, GNNs use a message-passing framework to iteratively aggregate and update node features. At each layer l, a node i aggragates messages from its neighbors j to compute its updated feature representation $h_i^{(l+1)}$ [2]:

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} [S]_{ij} W^{(l)} h_j^{(l)} + b^{(l)}\right), \tag{1}$$

where:

- $\mathcal{N}(i)$: The set of neighbors of node i.
- $W^{(l)}$: Trainable weight matrix at layer l.
- σ: Non-linear activation function.
- $b^{(l)}$: Bias term at layer l.

This iterative aggregation enables GNNs to capture both local node-level features and global graph-level properties, making them highly expressive.

Linear Shift-Invariant Filters and Graph Convolutions To draw parallels with signal processing, GNNs generalize the concept of convolution to graph structures. As described by Ruiz et al. [7], graph convolutions are defined using the GSO:

$$H(S)x = \sum_{k=0}^{K-1} h_k S^k x, \tag{2}$$

where h_k are filter coefficients, and S^k represents the k-hop neighborhood of the graph. This definition allows GNNs to model complex dependencies across multiple hops while maintaining a focus on the graph's inherent structure. The convolutional nature of GNNs is what allows them to process and learn from graph-structured data effectively.

4 Dataset

The MovieLens-100k dataset consists of 100,000 user ratings for 1,682 movies, provided by 943 unique users. Each rating, ranging from 1 to 5, reflects how much a user enjoyed a particular movie. To construct the movie similarity graph W, we calculate pairwise similarities between movies using the training subset of the dataset. This is done by analyzing the normalized covariance of ratings for movies that were rated by the same users, capturing how closely two movies are perceived based on shared preferences.
To focus on the most relevant connections, we retain only the top 40 strongest similarities for each movie. The resulting graph W is then normalized to ensure consistency and ease of computation. To further compare and evaluate graphs, we also measure their differences using tree distance, which provides insights into the structural relationships between graph representations.

5 Methodologies

To address the limitations talked about in section 2.4, we propose two novel sampling strategies: leverage score sampling and adapted BFS sampling. Leverage score sampling prioritizes nodes based on their contribution to the graph's overall structure, ensuring that critical spectral and structural properties are retained in the sampled subgraph. Adapted BFS sampling combines the connectivity-preserving benefits of traditional BFS with a probabilistic component, reducing local biases and enhancing the representativeness of the reduced graph. Together, these methods provide a practical and theoretically grounded solution to the challenges posed by large-scale graphs, as demonstrated in our experiments.

5.1 Leverage Scores Sampling Method

The motivation for adopting leverage scores as a sampling method lies in their ability to quantify the importance of individual nodes in a graph. Derived from regression and linear algebra, leverage scores measure how much a node contributes to the overall structure of the graph. In essence, they help identify nodes that are most representative

of the graph's global properties. This is particularly critical in scenarios where reducing graph size is necessary for computational efficiency, but maintaining structural integrity is equally important.

In the context of our graph-based tasks, such as those built from the MovieLens-100k dataset, leveraging the covariance matrix to calculate edge weights aligns naturally with the use of leverage scores. Both tools emphasize structural relationships and the influence of nodes, making leverage scores an intuitive choice for sampling.

Algorithm 1 Leverage Scores Sampling Method

Require: Graph $G = (V, E, A)$, number of nodes to sample k
Ensure: Sampled subset of nodes V_{sampled}

$V_{\text{sampled}} \leftarrow \emptyset$
Compute matrix factorization $A = U\Sigma V^\top$
for each node $i \in V$ **do**
    Compute leverage score $l_i \leftarrow |u_i|^2$, where u_i is the i-th row of U
end for
Compute total leverage score $L_{\text{total}} \leftarrow \sum_{i \in V} l_i$
for each node $i \in V$ **do**
    Calculate sampling probability $p_i \leftarrow \frac{l_i}{L_{\text{total}}}$
end for
while $|V_{\text{sampled}}| < k$ **do**
    Randomly sample node i from V with probability p_i
    Add node i to V_{sampled}: $V_{\text{sampled}} \leftarrow V_{\text{sampled}} \cup \{i\}$
end while
return V_{sampled}

5.2 Adapted BFS Method

Sampling strategies for graphs aim to reduce graph size while preserving key structural properties. While BFS (Breadth-First Search) is a well-known traversal method, its direct application in graph sampling often introduces local biases, favoring highly connected nodes within the immediate neighborhood of the starting point. This can lead to poor representativeness of the global graph structure.

To address this, our Adapted BFS method builds upon the strengths of traditional BFS while introducing additional mechanisms—random walks and beam search—to balance local connectivity with global representativeness.

5.2.1 Traditional BFS

BFS is highly effective for exploring local neighborhoods in a graph. It systematically visits all neighbors of a node before moving to their neighbors, ensuring complete coverage of each "layer" of the graph. However, this focus on local exploration may fail to capture nodes or regions that are distant but structurally important to the overall graph.

Algorithm 2 BFS

Require: Predicted point P, target graph node count θ
Ensure: Subgraph containing θ nodes

Initialize current layer $\leftarrow P$
Initialize new graph $\leftarrow P$
while new graph node count $< \theta$ **do**
  for each node in the current layer **do**
    Retrieve all neighbors of the node
    Add neighbors to new graph until reaching θ nodes
  end for
  if new graph node count $< \theta$ **then**
    Update current layer $\leftarrow$ neighbors
  end if
end while
return new graph

5.2.2 BFS with Random Walk

Incorporating random walks into BFS introduces stochasticity into the traversal process. Instead of strictly visiting all neighbors of a node, the traversal includes probabilistic steps to explore less obvious paths. This randomness reduces local bias, allowing the sampling method to reach more diverse regions of the graph. It effectively balances exploration (visiting new regions) and exploitation (focusing on dense regions).

Algorithm 3 BFS with Random Walk

Require: BFS smaller graph, probability ϵ
Ensure: Optimized smaller graph

while not converged **do**
  Randomly select one node from the graph and one node outside the graph
  Swap the selected nodes and compute the new correlation matrix
  if new correlation is larger than the old correlation **then**
    Accept the change
  else
    Accept with probability $\log(\text{old correlation}) - \log(\text{current correlation}) < \epsilon$
  end if
end while
return Optimized smaller graph

5.2.3 Beam Search

Beam Search enhances BFS by prioritizing nodes based on a scoring function, such as node degree or connectivity strength. At each step, only the most promising nodes are expanded, limiting the search to paths that are likely to yield the most representative

nodes. This makes the sampling process more efficient while still retaining the most structurally significant regions of the graph.

Algorithm 4 Beam Search

Require: Predicted point P, target graph node count θ, beam size k
Ensure: Subgraph containing θ nodes

```
Initialize current layer ← P
Initialize new graph ← P
while new graph node count < θ do
    Retrieve all neighbors of the current layer nodes
    Add up to k neighbors with the highest degree in the original graph to the new graph
    if new graph node count < θ then
        Update current layer ← top k neighbors
    end if
end while
return new graph
```

6 Experiments

In this experiment, we sought to evaluate how effectively leverage scores sampling and BFS-based sampling could preserve the structural integrity and predictive performance of the movie similarity graph. Our focus was on predicting the ratings of the top-1, top-2, and top-5 movies using a two-layer GNN trained on nodes sampled via these methods.

The process involved ranking nodes by their leverage scores and sampling them with BFS separately to generate training subsets. During the training phase, we performed a comprehensive grid search to identify the optimal combination of batch size, GNN weights, learning rate, epochs, and other hyperparameters. This ensured the GNN models were fine-tuned for each specific sampling strategy and prediction task.

After training, the GSO was transferred back to the larger graph to evaluate the model's performance on the test set. To ensure robust results, we repeated the experiments ten times and reported the average rMSE (root Mean Squared Error) for each sampling method across the prediction tasks. This allowed us to directly compare the performance of leverage scores sampling and BFS methods for predicting ratings of one, two, or five movies.

The experimental pipeline is illustrated in Figure 1, showcasing the workflow from graph sampling to final performance evaluation.

- **Top-1 Movie Prediction** Predicting the rating for a single movie is the most straightforward task. The GNN benefits from a highly focused target and achieves excellent predictive accuracy due to the limited scope of the prediction.
- **Top-2 Movies Prediction** Expanding to two target movies increases the prediction complexity. The GNN must balance information from the graph's structure to

simultaneously predict two ratings. This task demonstrates how well the GNN generalizes beyond a single target.

- **Top-5 Movies Prediction** Predicting ratings for five movies is a challenging task. The GNN must aggregate and process a broader range of graph features to predict multiple ratings accurately. While this approach increases computational cost, it showcases the GNN's capacity for handling complex multi-target predictions.

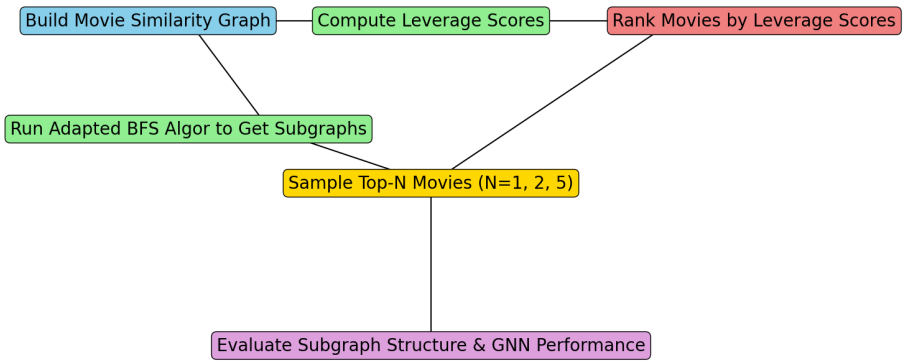

Figure 1: Pipeline: Experiments for Top-N Movies

7 Results and Discussion

7.1 Leverage Scores Sampling Method

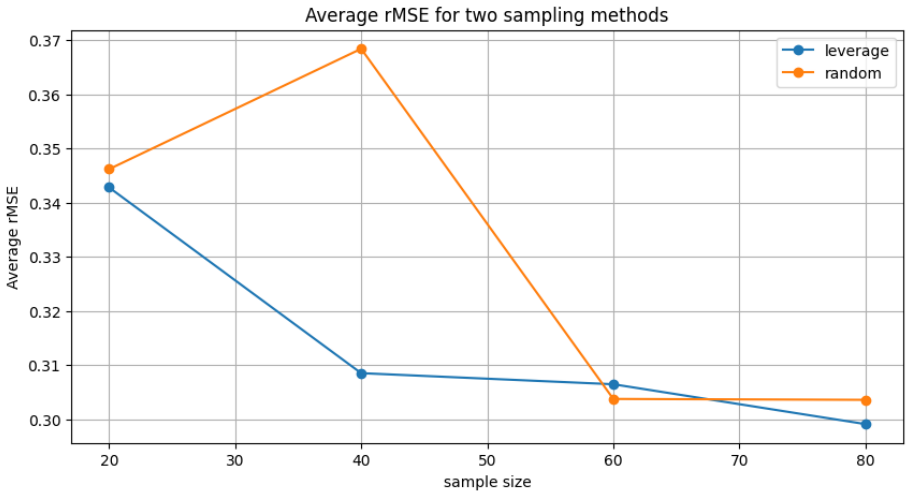

Figure 2: Top 1 Movie Prediction Average rMSE scores in 10 iterations got from test set on the larger graph. The orange one is random sampling and the blue one is the leverage scores sampling method

Top 1 Movie According to Figure 2, when predicting the rating of a single movie, the leverage scores sampling method consistently outperforms random sampling in terms of average rMSE across sample sizes of 20%, 40%, and 80%. The lowest average rMSE, approximately 0.3, is achieved with a sample size of 80%, demonstrating the effectiveness of leverage scores sampling in retaining the structural integrity of the graph and facilitating accurate predictions.

Furthermore, as shown in Figure 3, leverage scores sampling exhibits significantly lower variability in rMSE across 10 iterations compared to random sampling. This indicates a more stable and reliable pattern, highlighting the robustness of leverage scores sampling in producing consistent results, even in iterative scenarios. These observations emphasize its suitability for tasks requiring high precision and stability in graph-based predictions.

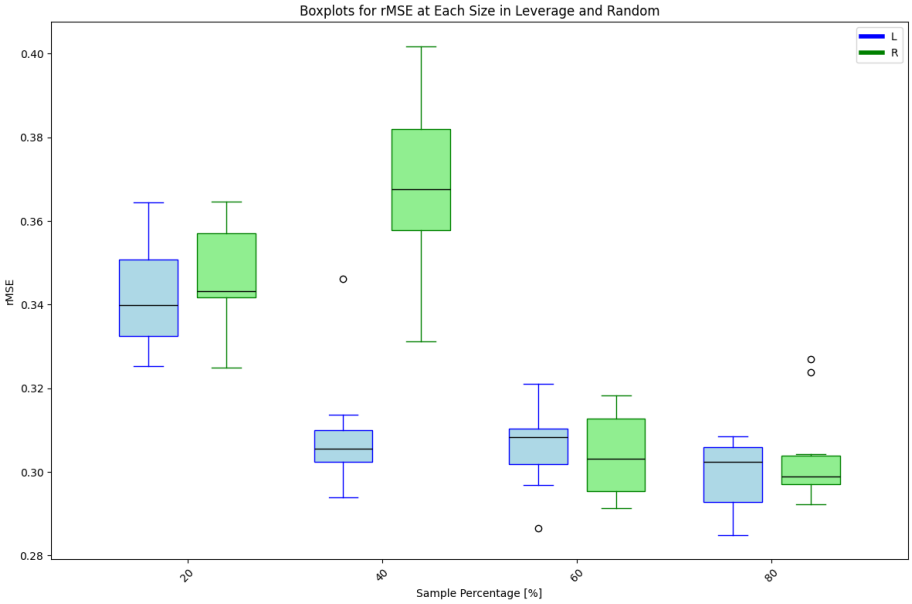

Figure 3: Top 1 Movie Prediction rMSE scores boxplots for 10 iterations got from test set on the larger graph. The green one is random sampling and the blue one is the leverage scores sampling method

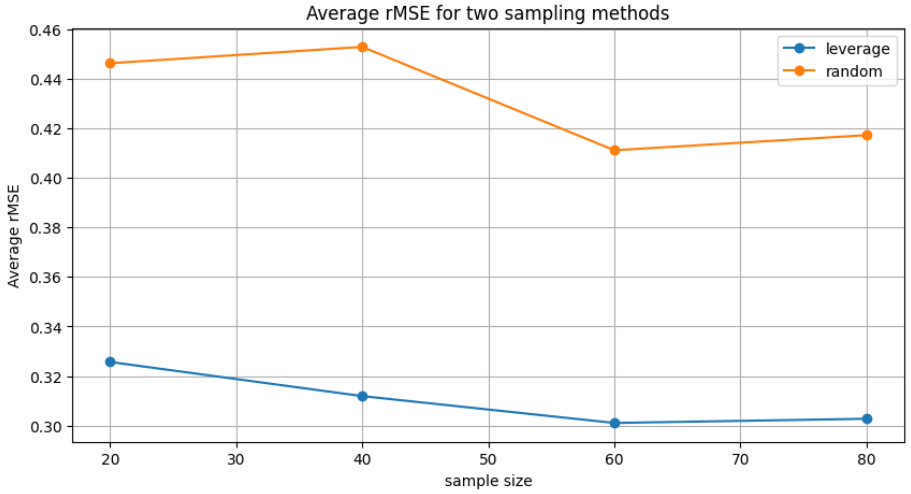

Figure 4: Top 2 Movie Prediction Average rMSE scores in 10 iterations got from test set on the larger graph. The orange one is random sampling and the blue one is the leverage scores sampling method

Top 2 Movies According to Figure 4, when predicting the ratings of two movies, the leverage scores sampling method demonstrates superior performance compared

to random sampling, both in terms of accuracy and stability. It consistently achieves lower average rMSE scores across all sample sizes. Notably, the lowest average rMSE, approximately 0.3, which is about 26.8% is observed at a sample size of 60%, further emphasizing the efficiency of leverage scores sampling in preserving essential graph structure for predictive tasks.

As shown in Figure 5, similar to the top-1 movie prediction task, leverage scores sampling also exhibits significantly lower deviations in rMSE across 10 iterations, coupled with fewer outliers compared to random sampling. This indicates a more stable and reliable performance pattern. These results highlight the robustness of leverage scores sampling, making it a more effective choice for multi-target prediction tasks where both accuracy and consistency are crucial.

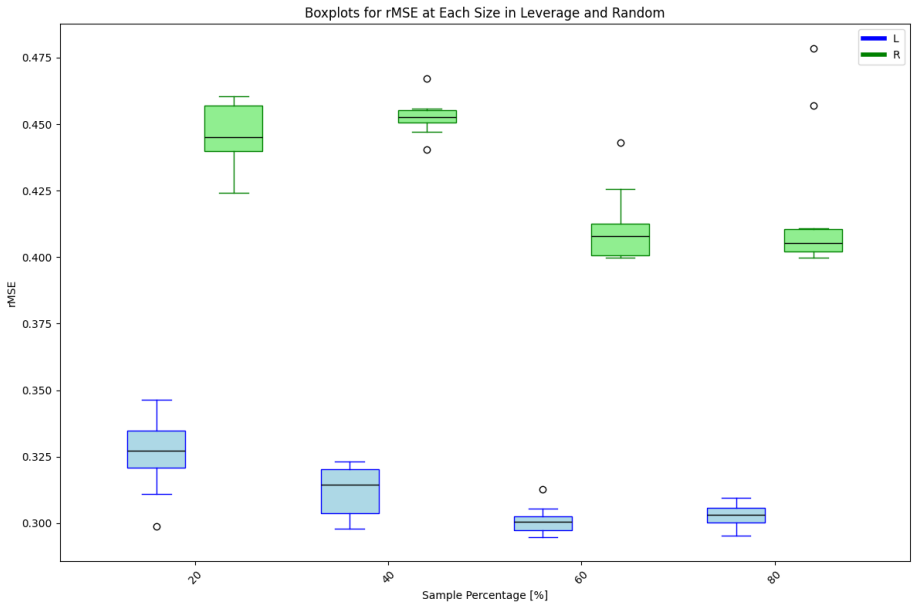

Figure 5: Top 2 Movie Prediction rMSE scores boxplots for 10 iterations got from test set on the larger graph. The green one is random sampling and the blue one is the leverage scores sampling method

Top 5 Movies However, as observed in Figure 6, when the prediction task is expanded to five movie ratings, the results deviate significantly from the patterns observed in the top-1 and top-2 movie prediction tasks. In this scenario, the leverage scores sampling method underperforms compared to random sampling, achieving notably higher average rMSE scores across most sample sizes. The only exception is at a sample size of 40%, where leverage scores sampling yields a slightly lower average rMSE. At larger sample sizes, such as 80%, the leverage scores sampling method performs poorly, with average rMSE reaching approximately 0.6.

Similarly, as shown in Figure 7, leverage scores sampling exhibits greater variability in rMSE across 10 iterations, with higher deviations and an increased number of outliers

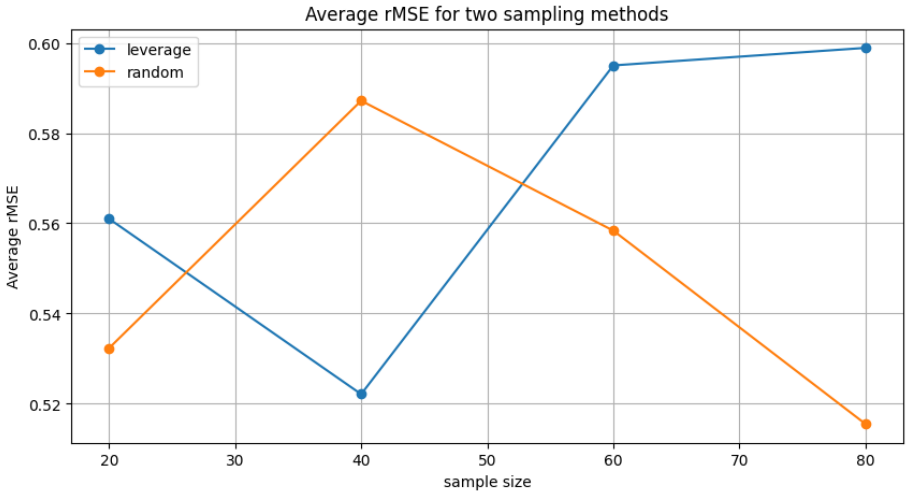

Figure 6: Top 5 Movie Prediction Average rMSE scores in 10 iterations got from test set on the larger graph. The orange one is random sampling and the blue one is the leverage scores sampling method

compared to random sampling. This is particularly evident across all sample sizes except 40%. These results suggest that, unlike the earlier tasks, leverage scores sampling lacks the robustness required for multi-target prediction tasks involving a larger number of movies. The instability and reduced performance highlight the limitations of leverage scores sampling when applied to more complex prediction scenarios, where the structure of the sampled subgraph may no longer adequately preserve the global properties needed for accurate predictions.

Comparison When the number of predicted movie ratings (n) is small, such as in the top-1 and top-2 prediction tasks, the leverage scores sampling method consistently outperforms random sampling. This advantage stems from the method's ability to prioritize nodes with the highest leverage scores, which are the most structurally influential in the graph. By focusing on these high-impact nodes, leverage scores sampling preserves critical relationships within the subgraph, ensuring that the predictive model has access to the most relevant structural information. Additionally, this method demonstrates higher stability, as evidenced by lower deviations and fewer outliers in rMSE across iterations. These findings suggest that leverage scores sampling is particularly well-suited for tasks involving a small number of predictions, such as personalized recommendations, where capturing the most critical regions of the graph is sufficient.

However, when the number of predicted ratings increases ($n \geq 5$), the performance of leverage scores sampling declines, and random sampling begins to outperform it, particularly at larger sample sizes. This could be due to the fact that not all predicted nodes have high leverage scores, and their neighbors, which might also lack high leverage scores, may be excluded from the sampled subgraph. Consequently, some regions of

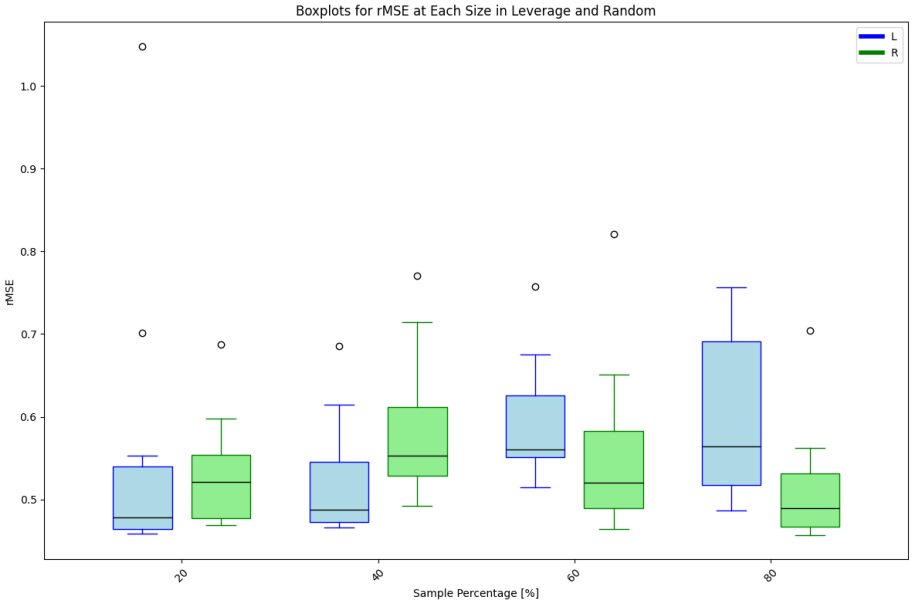

Figure 7: Top 5 Movie Prediction rMSE scores boxplots for 10 iterations got from test set on the larger graph. The green one is random sampling and the blue one is the leverage scores sampling method

the graph that are essential for accurate predictions are poorly represented, leading to suboptimal results. Moreover, as the sample size grows, random sampling increasingly approximates the structure of the full graph, preserving more diverse connections and outperforming the localized approach of leverage scores sampling. This highlights a potential limitation of the leverage scores method, suggesting the need to evaluate whether the predicted nodes themselves and their neighborhoods are adequately captured in the sampling process.

Interestingly, at a sample size of 40%, leverage scores sampling achieves consistently strong performance with low average rMSE and minimal variability across all tasks. This suggests that 40% might represent an optimal balance between retaining high-leverage nodes and maintaining sufficient structural diversity. Smaller sample sizes, such as 20%, might miss critical regions of the graph, while larger sample sizes, like 80%, may introduce redundancy, reducing the method's overall effectiveness. This observation indicates that the effectiveness of leverage scores sampling is task- and sample size-dependent, warranting further exploration into how sample size impacts its performance. Testing intermediate sample sizes and analyzing the distribution of leverage scores within subgraphs could provide deeper insights into optimizing the method for different predictive tasks.

7.2 Adapted BFS Methods

In this work, BFS, Beam Search, and BFS Random Walk were evaluated, focusing primarily on the top-1 movie prediction task. BFS demonstrated moderate performance, with MSE improving from 0.342 to 0.323 as the sample size increased. Its neighborhood-focused sampling preserved local structural patterns, such as clusters, leading to better predictive accuracy compared to random sampling. However, for more complex tasks, such as top-5 movie predictions, BFS underperformed due to its limitations in capturing broader graph connectivity.

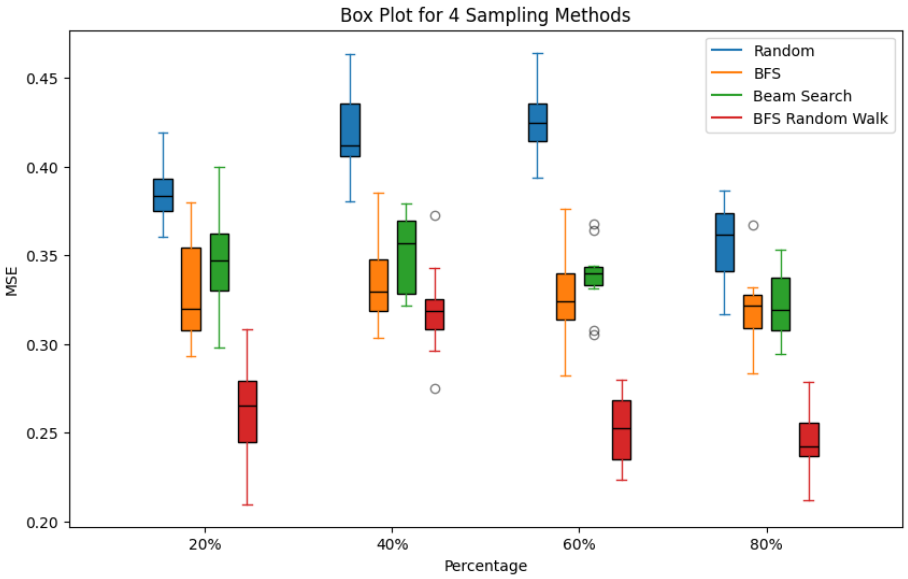

Figure 8: Top 1 Movie Prediction rMSE scores boxplots for 10 iterations got from test set on the larger graph for Adapted BFS Methods.

Beam Search, while showing some improvement over random sampling, performed less effectively than BFS. MSE dropped from 0.356 to 0.326 as the sample size increased, but the method struggled with diminishing returns at larger sample sizes. This was attributed to its path-based approach, which prioritized sequential relationships over local structural fidelity, limiting its ability to represent complex graph structures.

BFS Random Walk emerged as the most effective approach, achieving the lowest MSE values, ranging from 0.315 to 0.249. By combining BFS's neighborhood continuity with the exploratory nature of random walks, this method balanced local and global structural preservation, outperforming all other methods. However, its performance showed variability at smaller sample sizes, likely due to insufficient random walks to effectively capture the graph's complexity. This variability highlights the need for careful parameter tuning, particularly for tasks requiring broader graph coverage.

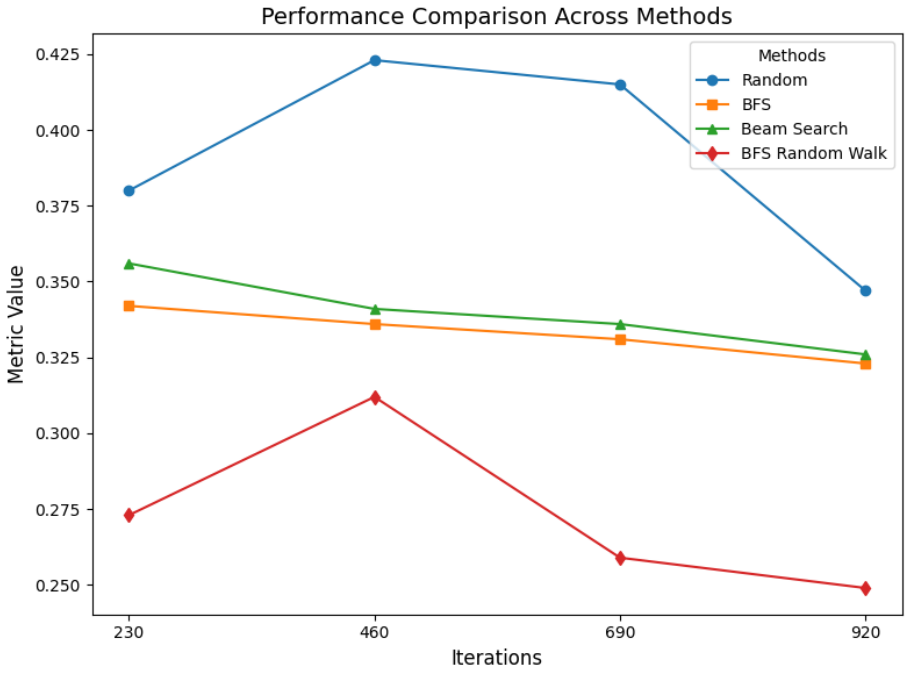

Figure 9: Top 1 Movie Prediction Average rMSE scores in 10 iterations got from test set on the larger graph with Adapted BFS Methods

Overall, while BFS and Beam Search showed potential for localized predictions, BFS Random Walk stood out for its superior ability to maintain structural fidelity and achieve accurate predictions, particularly in larger sampling scenarios.

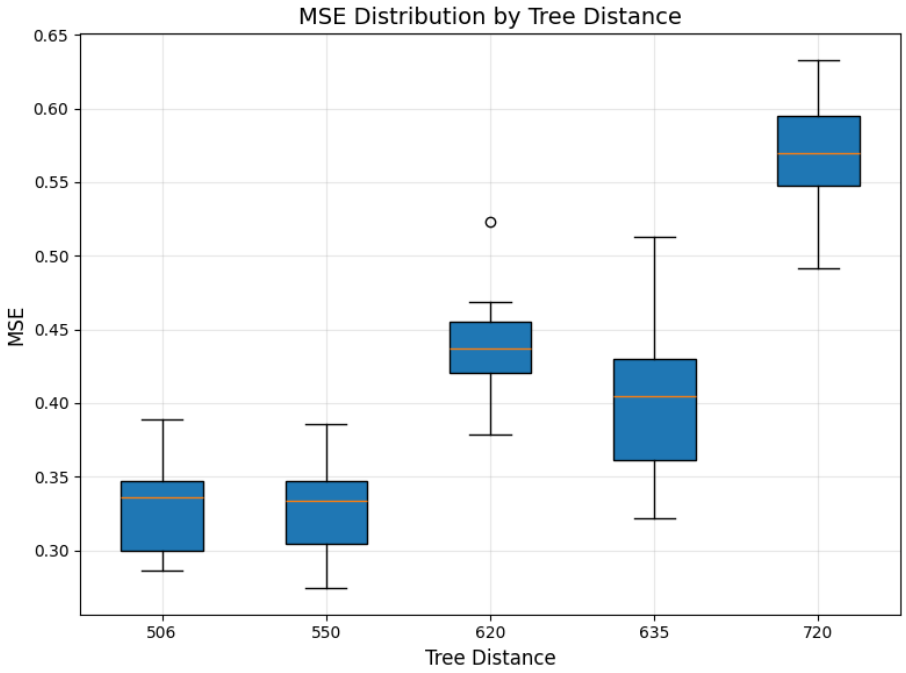

Figure 10: rMSE Distributions on Tree Distance

8 Limitations and Future Work

This work is constrained by several limitations that leave room for future exploration. First, the experiments were conducted using a two-layer GNN, which, while effective, was chosen to balance computational cost. Extending to deeper GNN architectures could potentially yield better performance, albeit at a higher computational expense. Second, the dataset used—MovieLens-100k, with 943 users and 1,682 movies—is relatively small. Larger, more diverse datasets could provide a better understanding of the scalability and generalizability of the sampling methods. Finally, both leverage scores sampling and BFS-based methods demonstrated a decline in performance as the number of movies predicted simultaneously (n) increased. This highlights a key limitation: neither method is fully optimized for multi-target prediction tasks, particularly when n becomes large, indicating that further refinement is needed.

To address these limitations and improve the overall performance, several promising directions for future work are proposed. First, hybrid sampling methods could be explored, combining leverage scores and random sampling to balance the retention of critical nodes with broader graph coverage. This approach might enhance performance, particularly for larger n, by capturing both local and global graph structures. Second, examining the distribution patterns of the graph in the context of the prediction task could uncover whether certain sampling methods perform better at specific sample sizes. Such insights could inform adaptive sampling strategies tailored to the graph's

characteristics. Finally, developing a mathematical derivation of convergence for the sampling methods could provide a theoretical foundation for their performance. This could lead to better parameter tuning and an understanding of when and why certain methods succeed or fail, guiding the design of more robust sampling techniques.

9 Conclusion

This study examined the effectiveness of leverage scores sampling and BFS-based methods in graph sampling for predicting movie ratings using Graph Neural Networks (GNNs). Our findings reveal that leverage scores sampling excels in small-scale prediction tasks ($n < 5$), achieving lower average rMSE and greater stability compared to random sampling. However, as the number of predicted movies increases ($n \geq 5$), its performance declines, particularly at larger sample sizes, where random sampling and BFS-based methods demonstrate better results. This highlights the limitations of leverage scores sampling in capturing the diverse structural relationships necessary for more complex tasks.

Among the sampling methods tested, BFS Random Walk consistently achieved the best performance, combining the localized structural benefits of BFS with the exploratory advantages of random walks. This hybrid approach allowed for more comprehensive graph representation, resulting in the lowest MSE values across varying sample sizes. In contrast, Beam Search and BFS showed moderate improvements but faced diminishing returns, particularly in larger sampling scenarios.

Moving forward, this work underscores the importance of exploring hybrid sampling techniques, such as combining leverage scores with random sampling, to address the challenges of multi-target prediction tasks. Future research should also focus on leveraging larger datasets, testing deeper GNN architectures, and developing theoretical insights, such as convergence guarantees, to further refine these methods. By addressing these limitations, we can unlock the full potential of graph sampling in enhancing the scalability, robustness, and predictive accuracy of GNNs for real-world applications.

References

[1] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

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

[3] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems* (D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds.), vol. 29, Curran Associates, Inc., 2016.

[4] L. Ruiz, F. Gama, A. G. Marques, and A. Ribeiro, "Invariance-preserving localized activation functions for graph neural networks," *IEEE Transactions on Signal Processing*, vol. 68, pp. 127–141, 2020.

[5] L. Ruiz, L. F. O. Chamon, and A. Ribeiro, "The graphon fourier transform," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5660–5664, 2020.

[6] L. Ruiz, L. F. O. Chamon, and A. Ribeiro, "Graphon neural networks and the transferability of graph neural networks," 2020.

[7] L. Ruiz, L. F. O. Chamon, and A. Ribeiro, "Transferability properties of graph neural networks," *Trans. Sig. Proc.*, vol. 71, p. 3474–3489, July 2023.

[8] J. Leskovec and C. Faloutsos, "Sampling from large graphs," KDD '06, (New York, NY, USA), p. 631–636, Association for Computing Machinery, 2006.

[9] "Network sampling designs for relational classification," vol. 6, pp. 383–386, Aug. 2021.

[10] M. W. Mahoney and P. Drineas, "Cur matrix decompositions for improved data analysis," *Proceedings of the National Academy of Sciences*, vol. 106, no. 3, pp. 697–702, 2009.