Shapley-Value-Based Graph Sparsification for GNN Inference

Selahattin Akkas
Department of Intelligent Systems Engineering
Indiana University Bloomington
Bloomington, Indiana, USA
sakkas@iu.edu

Ariful Azad
Department of Computer Science & Engineering
Texas A&M University
College Station, Texas, USA
ariful@tamu.edu

ABSTRACT

Graph sparsification is a key technique for improving inference efficiency in Graph Neural Networks by removing edges with minimal impact on predictions. GNN explainability methods generate local importance scores, which can be aggregated into global scores for graph sparsification. However, many explainability methods produce only non-negative scores, limiting their applicability for sparsification. In contrast, Shapley value based methods assign both positive and negative contributions to node predictions, offering a theoretically robust and fair allocation of importance by evaluating many subsets of graphs. Unlike gradient-based or perturbation-based explainers, Shapley values enable better pruning strategies that preserve influential edges while removing misleading or adversarial connections. Our approach shows that Shapley value-based graph sparsification maintains predictive performance while significantly reducing graph complexity, enhancing both interpretability and efficiency in GNN inference.

1 INTRODUCTION

Graph Neural Networks (GNNs) have become very popular in the field of machine learning, specifically in handling graph-structured data [35]. Unlike traditional neural networks, GNNs leverage the hidden relationships within graph data, making them one of the preferred methods for social networks [11, 18], recommendation systems [6, 33], molecular modeling [8, 35], and financial fraud detection [4, 19]. GNNs' ability to capture local and global patterns through message passing has significantly improved predictive model performance in these domains [46].

Since the weight matrices in GNNs are typically small, the computational and memory complexity of GNN inference is often dominated by the size of the graph, that is, the number of nodes and edges. As real-world graphs continue to grow in size and structural complexity, the scalability and feasibility of GNN inference on memory-constrained edge devices and GPUs become increasingly challenging. To address this issue, various graph sparsification techniques [3, 7, 28, 38, 41] have been introduced in the literature. Most of these methods are inspired by the Lottery Ticket Hypothesis, which suggests that a subnetwork of a GNN consisting of a subset of parameters, layers, nodes, and/or edges can be trained to achieve performance comparable to that of the full model. These approaches, therefore, focus on identifying and pruning redundant edges to reduce graph complexity, lower resource consumption, and accelerate inference.

A second class of methods also seeks to identify uninformative nodes and edges, but with the goal of explaining the predictions made by GNN models [5, 14, 32, 37]. The GNN explanation methods aim to identify crucial subgraphs contributing more to the predictions. The fidelity [39] metric is commonly used to evaluate the success of GNN explanation methods. It measures how model predictions change when some edges are removed. Specifically, $Fidelity_+$ measures how the model prediction changes when important edges are removed, while $Fidelity_-$ measures how the model prediction changes when the least important edges are removed. The ability to identify the most and least important edges for GNN explanation methods motivated us to apply explanation scores to graph sparsification. Graph sparsification using explanation scores offers several advantages: (1) it eliminates the need to retrain the model after sparsification, (2) it avoids the need to repeatedly recompute edge scores to determine the optimal sparsity level, and (3) the inference is inherently explainable as the graph is already sparsified but removing unimportant edges.

While the fidelity score is a good metric for evaluating GNN explanation methods, it has limitations. The fidelity metric only focuses on the magnitude of score change in the model prediction; it does not consider whether the model prediction improves or worsens. Many GNN explanation methods generate only non-negative explanation scores to define importance. However, some edges significantly reduce model prediction, but those edges are considered important since they significantly change the fidelity score. Explanation methods that only give non-negative scores have this shortcoming. We argue that an explanation score that can provide positive and negative importance scores should perform similarly or better on graph sparsification tasks.

Shapley value based GNN explanation methods [1, 17, 22, 40] provide both positive and negative attribution scores in addition to high-quality explanations. Figure 1 shows an example of Shapley values for a node. In the figure, the edge from node 37 to node 60 is the most important edge; however, removing this edge improves model performance. The ability to prune the graph's negatively attributing and least significant edges without compromising the accuracy allows for higher sparser graphs and faster inference.

In this paper, we explore graph sparsification using Shapley values and demonstrate its unique advantages over existing explanation methods for node classification tasks, particularly in reducing the computational complexity of inference. Overall, the paper makes the following contributions:

- We comprehensively evaluate Shapley value explanations across multiple datasets and models, demonstrating its robustness and generalizability on graph sparsification.
- We compare state-of-the-art GNN explanation methods and sparsification techniques, highlighting superior performance in maintaining model accuracy.
- We also compare our sparsification approach with graph lottery ticket approaches, demonstrating competitive or

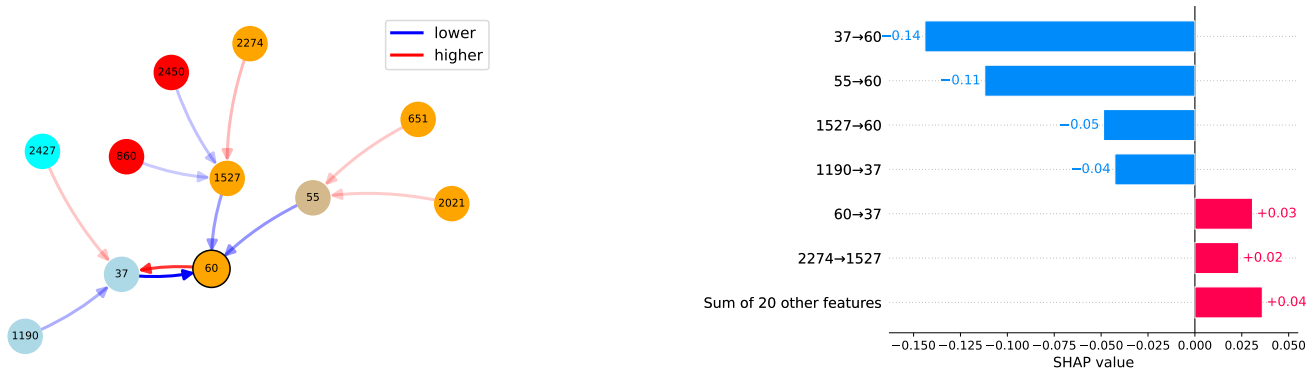

Figure 1: Example Shapley value explanation on Cora node 60. Node colors denote classes. The bar chart on the right shows important edges and their Shapley values. While red colors show a positive contribution, blue colors show a negative contribution.

improved sparsification ratios while maintaining model performance.

2 BACKGROUND & RELATED WORK

We denote a graph as $G = \{V, E\}$, where V is the set of N nodes, E is the set of edges, and $X \in \mathbb{R}^{N \times F}$ is the node feature matrix. $A \in \{0,1\}^{N \times N}$ is the binary adjacency of the graph, where $A_{ij} = 1$ if $v_i, v_j \in E$, and $A_{ij} = 0$ otherwise. Let $y = \{y_1, y_2, ..., y_N\}$ denote the labels of the nodes, where each label y_i belongs to one of $\mathbb{C}$ classes in a multiclass node classification task. An l-layered GNN model f takes X and A as input and generates predictions for the ith node: $\hat{y_i} = f(X, A)$, where $\hat{y_i} \in \mathbb{C}$.

Computational graph. When predicting the class of a node, the GNN inference needs a small subgraph of the entire graph. Specifically, the prediction of a node v with an l-layer GNN only depends on v's 1-hop through l-hop neighbors, the edges among them, and any associated node and edge features. This l-hop subgraph is referred to as the computational graph $G_c(v)$, which contains all the necessary information for predicting v.

2.1 GNN Explanations

A GNN explanation method Φ generates explanations for a given node v with respect to a target class $t \in \mathbb{C}$. The target class may correspond to either the ground truth label or the predicted class. Popular GNN explanation methods aim to explain the prediction for node v by taking as input a trained GNN model f and the node's computational graph $G_c(v)$. The explanation typically consists of a small subgraph $G_s(v) \subseteq G_c(v)$ and/or a subset of node features that most significantly influence the prediction. The key idea is to retain only the edges and features that contribute most to the model's decision. In this work, we focus exclusively on subgraph-based explanations, as our goal is to sparsify graphs by pruning edges. The explanation model assigns an importance score $\phi_v^t(i, j)$ to each edge (v_i, v_j), indicating its contribution to node v's prediction for the target class t. Depending on the explanation method, these scores can be either positive or negative.

2.2 Related Work

Graph sparsification aims to remove edges from a graph while preserving the model's predictive performance. Various approaches have been proposed in the literature to achieve this goal.

Denoising methods, such as NeuralSparse [45] and PTDNet [15], aim to enhance the GNN's generalization capability by reducing noise and making the GNN less sensitive to the graph's quality. NeuralSparse learns irrelevant edges during training and removes them to improve node representations, whereas PTDNet utilizes a probabilistic edge dropout mask to learn noisy edges and subsequently drops them.

Graph lottery ticket approaches aim to find a sparser graph and model parameters with similar or better accuracy than the original graph and model, which are called winning tickets. UGS [3], Early-Bird GCNs [38], WD-GLT [7], CGP [12], ICGP [28], FastGLT [41] and [16, 42–44] aim to find the winning tickets. While these approaches can sparsify graph and model parameters, shallow GNN models (e.g., 2-4 layers) usually work well. Moreover, most of these approaches require retraining for each target sparsity level. Since models are shallow, starting with a small model and experimenting with larger or deeper models is more practical. In addition, they only use a small portion of nodes (the training set) in semi-supervised GNN learning, which has a limited impact on graph sparsification.

Explainability based graph sparsification approaches use explanation scores to sparsify the graph. EEGL [20] applies frequent subgraph mining to find the most common patterns using GNNExplainer. Then, patterns are used as additional features. The authors train the model iteratively with found subgraphs to improve the model's accuracy. While EEGL finds subgraphs, it mainly focuses on enhancing the model's performance rather than sparsification.

IGS [10] focuses on brain graph sparsification. It iteratively learns a trainable edge mask during training and removes unimportant edges. xAI-Drop [13] computes node explanations and drops nodes based on the explainability score. It first considers the model's probability for candidate nodes. Then, it computes explainability scores. Finally, it applies Bernoulli-based node drops. While these works utilize explainability, their primary target is to enhance model accuracy during training. We focus on enhancing the inference speed of the pre-trained model.

The work by Shin et al. [26] is closely related to ours, as it leverages GNN explanations for graph sparsification. The authors propose a fidelity-inspired pruning method, where the $Fidelity_{-}$ score measures the change in model prediction when non-essential edges are removed. They aggregate individual edge explanation scores into global importance scores and prune edges with the lowest values. However, their method considers only non-negative edge scores. While $Fidelity_{-}$ effectively identifies unimportant edges, we show that a better sparsification can be achieved by removing edges with negative importance (i.e., edges that reduce prediction confidence).

3 METHOD

3.1 Shapley Values

Shapley value [24] is a game-theoretic method that fairly distributes gains to collaborating players. A Shapley value GNN explanation method considers nodes or edges as **players** and fairly distributes the model output to players.

The exact Shapley value of a player is computed by Eq. 1, where n denotes the number of players, S a coalition (a subset of players), and $f(S\cup\{i\})-f(S)$ is player i's marginal contribution to coalition S. Shapley values can be positive and negative; while positive values increase model prediction probability, negative values decrease.

$$\phi_i = \sum_{S\subseteq\{1,\dots,n\}\setminus\{i\}}^{2^{n-1}} \frac{|S|!(n-|S|-1)!}{n!}\left[f(S\cup\{i\})-f(S)\right] \quad (1)$$

Computing exact Shapley values is impractical when the number of players is large, as it requires evaluating 2^n coalitions. [1, 5] use a simple surrogate model to compute the approximation of Shapley values using a much smaller subset of coalitions ($k \ll 2^n$). The surrogate model g is defined in Eq. 2, where $m \in \{0,1\}^{1xn}$ denotes a binary coalition mask, S, and ϕ are model parameters: the approximation of the Shapley values.

$$f(x) \approx g(x) = \phi_0 + \sum_{i=1}^{n} \phi_i m_i, \quad (2)$$

3.2 Graph Sparsification by Explanation Scores

Most GNN explanation methods give local explanations, i.e, an explanation for a node's classification. Since the same edges are in many different nodes' $l-hop$ neighborhoods, there are multiple scores for each edge. To get a global score for each edge, we need to aggregate scores. In this work, we use **mean** aggregation, where we calculate the average score for each edge. We show our sparsification algorithm in Algorithm 1 and Figure 2.

We also considered the sum and weighted mean aggregations. We use model predictions as probabilities in the weighted mean, thinking they should have less weight when the model is less sure. However, we don't see significant differences in pruning performance. The sum and weighted mean results are provided in Appendix A.

Algorithm 1 GNN Explanation-Based Graph Sparsification

Require: Graph $G=(V,E)$, edge importance scores $S_v(e)$ for each node $v\in V$ and edge $e\in E$, sparsification threshold τ
Ensure: Sparsified graph $G'=(V,E')$

```
Initialize empty set E' ← ∅
Initialize edge score map S ← ∅
for each edge e ∈ E do
    S(e) ← (1/|V_e|) Σ_{v∈V_e} S_v(e)        ▷ V_e: nodes utilize edge e
end for
Sort edges e ∈ E by S(e) in descending order into list L
for each edge e in L do
    if |E'| < (1 − τ) · |E| then
        E' ← E' ∪ {e}
    end if
end for
return G' = (V, E')
```

Table 1: Dataset Summaries

Dataset	Nodes	Edges	Features	Classes
Cora	2708	10556	1433	7
CiteSeer	3327	9104	3703	6
PubMed	19717	88648	500	3
Coauthor-CS	18333	163788	6805	15

4 EXPERIMENTS

We hypothesize that Shapley-based explanation methods are particularly effective for graph sparsification. While several Shapley-based GNN explanation methods have been proposed, we use GNNShap [1], a recent method that has demonstrated superior performance compared to other approaches. In our experiments, we compare GNNShap-based sparsification with other explanation methods. We also compare Shapley-based sparsification with graph lottery ticket (GLT) baselines. In each experiment, we apply different sparsification methods to the graph, perform GNN inference on the test nodes, and report the resulting test accuracy. A sparsification method is considered more effective if it maintains high test accuracy despite sparsification of the graph. For GNN explanations, we compute explanations for each node for predicted classes and aggregate explainability scores. We repeat each experiment five times and provide the average results.

4.1 Datasets

In our experiments, we utilize three well-known real-world citation datasets: Cora, CiteSeer, and PubMed [36], as well as a coauthorship dataset: Coauthor-CS [25]. In citation datasets, nodes represent papers, and edges represent citations. Node features are bag-of-words vectors, the most common words in the documents. Coauthor-CS is a coauthorship graph where nodes are authors and edges show coauthorships. Its node features are keywords in the papers. Since there is no public train, validation, and test split for Coauthor-CS, we randomly sample 30 nodes from each for the training and validation set, while using the remaining for testing. Table 1 shows summaries of datasets.

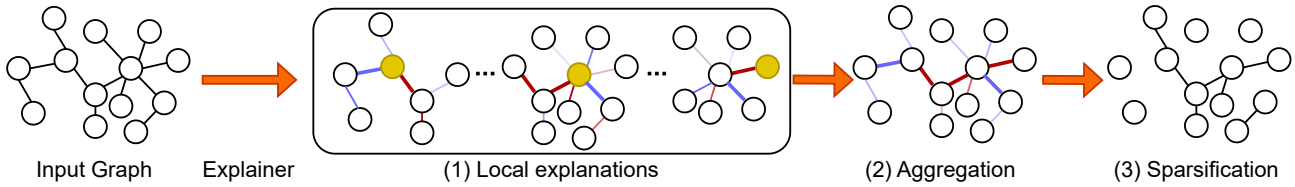

Figure 2: Overview of the explanation based graph sparsification algorithm. Firstly, explanation scores are computed for each node. Secondly, the scores are aggregated for each edge. Finally, using the aggregated scores, edges are pruned until the target sparsity ratio is reached.

Table 2: Trained GNN models and their training, validation, and test set accuracies.

Model	Dataset	Train	Validation	Test
GCN	Cora	100.00	79.40	81.50
	CiteSeer	99.17	70.40	71.00
	PubMed	100.00	80.20	78.80
	Coauthor-CS	95.33	93.33	91.99
GAT	Cora	100.00	79.40	81.40
	CiteSeer	99.17	73.20	71.60
	PubMed	98.33	80.60	78.50
	Coauthor-CS	94.22	92.22	91.29

4.2 Models

We use a two-layer GCN [9] and GAT [30] models with 16 hidden layer sizes for Cora, CiteSeer, and PubMed, and 64 for Coauthor-CS datasets. In the GAT models, we use eight attention heads. We use 0.5 dropout and ReLU as the activation function. We train the model for 200 epochs using a learning rate of 0.01. For GLT experiments, we follow UGS, using the same parameters with a 512 hidden layer size. Table 2 shows model training, validation, and test accuracies.

4.3 Baselines

4.3.1 GNN explanation methods.

- Saliency [2, 23]: uses the absolute values of gradients with respect to edges as scores.
- Guided Backpropagation [27]: also uses gradients, except negative gradients are pruned in the backpropagation.
- Integrated Gradients [29] computes the gradients of the model's output with respect to edges, tracing a path from a baseline to the actual input.
- GNNExplainer [37]: uses mutual information to learn edge scores. It uses a learnable mask and trains it iteratively using gradients to maximize the mutual information.
- PGExplainer [14]: also utilizes mutual information. It trains a neural network model to generate edge scores.
- FastDnX [21]: utilizes linear surrogate model based on the SGC [34] to explain GNN models.
- GraphSVX [5] is a Shapley value based GNN explanation method that uses a linear surrogate model to approximate Shapley values.
- GNNShap [1] is another Shapley value method specifically designed for GNNs. While it is similar to GraphSVX, it generates explanation scores for edges and utilizes a GPU for coalition sampling and model predictions. Therefore, it is an order of magnitude faster than GraphSVX and can evaluate more coalition samples.

FastDnX and GraphSVX generate scores for nodes. We convert node scores to edges by averaging the scores of connected edges.

While there are other Shapley value based GNN explanation methods, GraphShap [22] and EdgeSHAPEr [17] are designed for graph classification; SubGraphX [40] can only be used for small graphs. Therefore, we utilize GraphSVX and GNNShap in our work as two representative Shapley value-based GNN explanation methods.

4.3.2 Graph lottery ticket baselines.

- Unified GNN Sparsification (UGS) [3]: iteratively prunes the GNN model and the adjacency matrix to find a smaller model and graph that gives similar or higher accuracy. Then, it trains the pruned model with the pruned adjacency matrix. We disable model pruning and use the same parameters provided in the source code for UGS.
- WD-GLT [7]: addresses the limitation of UGS, which only considers a fraction of the adjacency matrix in the loss. WD-GLT adds the Wasserstein Distance (WD) [31] between nodes predicted to be in the same class to the loss.
- FastGLT [41]: proposes a one-shot pruning method as a faster alternative to iterative pruning, achieving higher sparsity and faster speeds. It starts with a pre-trained model and removes model weights and edges in a single step based on their magnitude. Finally, a denoising process is applied to minimize the effect of noise introduced during pruning.

5 RESULTS

5.1 Experimental Results with Explanation Methods

We present our graph sparsification results using various explanation methods in Figure 3. The figure demonstrates that the Shapley value-based method, GNNShap, consistently achieves higher test accuracy at high sparsification levels. For example, on the Cora dataset using both GCN and GAT models, GNNShap can prune 80% of the edges with less than a 2% drop in accuracy. Similarly, on the PubMed dataset with the GCN model, GNNShap can prune 80% of the edges with less than a 2% drop in accuracy. Notably, on PubMed and Coauthor-CS with the GAT model, GNNShap matches

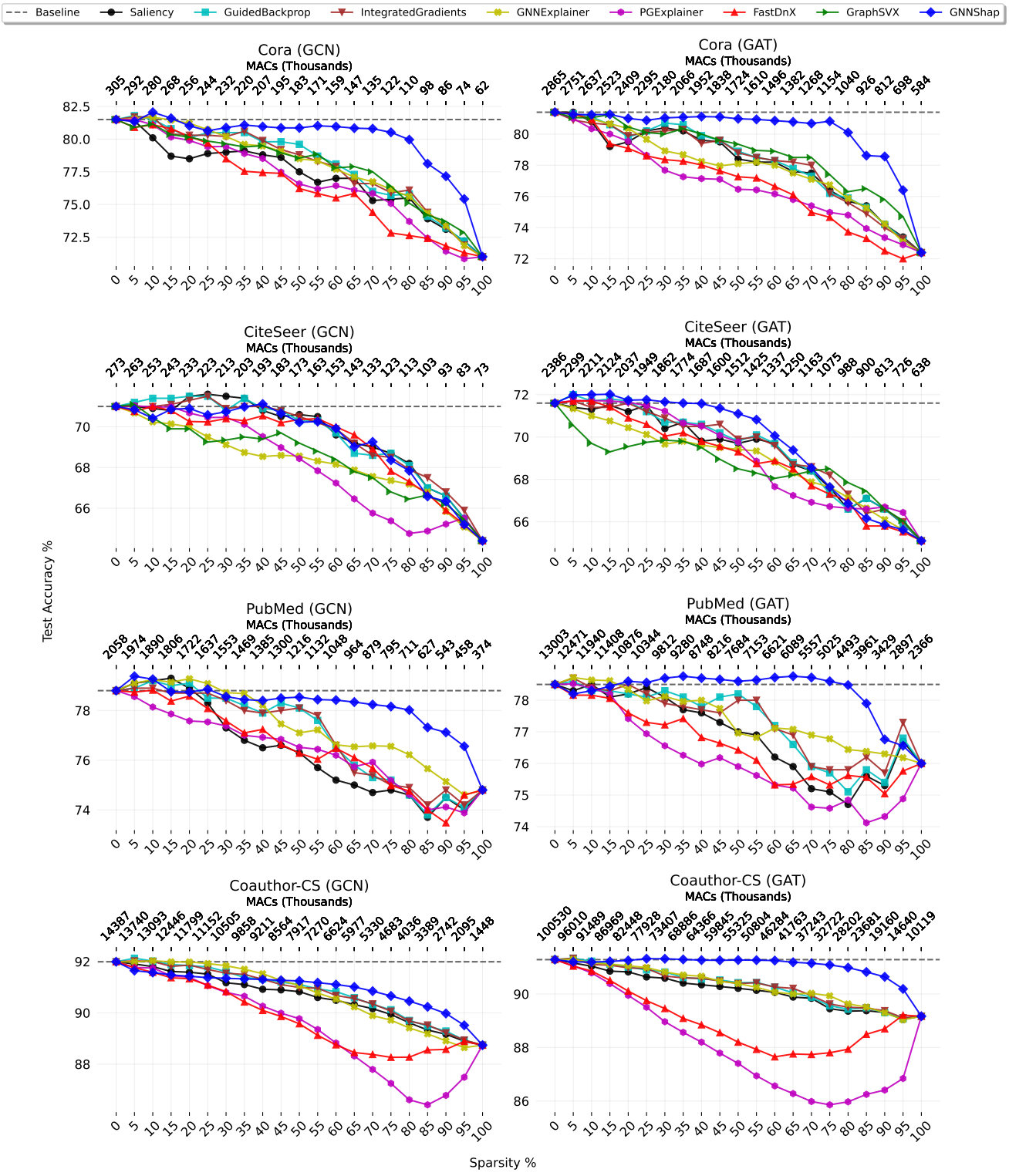

Figure 3: Test accuracies when edges sparsified using mean aggregated explanation scores. GNNShap gives competitive or even better accuracies for high sparsification percentages.

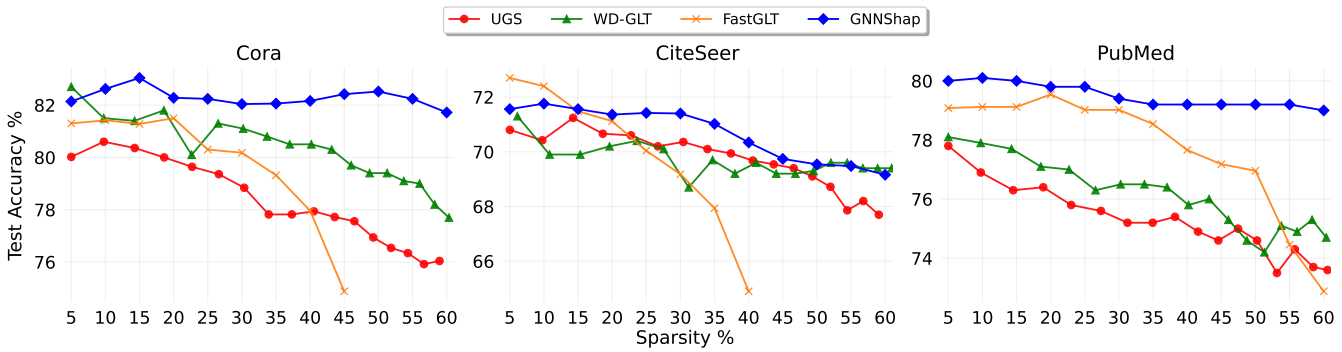

Figure 4: Test accuracy comparison of two-layer GCN model with 512 hidden layer size. Shapley value-based sparsification achieves higher accuracy with significantly less loss in accuracy.

the original accuracy even after pruning 80% and 55% of the edges, respectively.

Overall, GNNShap enables significantly higher pruning rates with minimal accuracy loss compared to other explanation methods. One notable exception is the CiteSeer dataset with the GCN model, where all explainers perform similarly. This is likely due to the lower test accuracy of the model on CiteSeer. GNNShap's superior performance can be attributed to (i) its ability to better distinguish between important and unimportant edges, and (ii) its capacity to assign both positive and negative attribution scores to edges.

To evaluate the computational efficiency of Shapley value-based sparsification, we report the number of Multiply-Accumulate operations (MACs) required during the message-passing step of GNN inferences. For example, on the Cora dataset using a GCN model, inference on the original graph requires 305,000 MACs. With 80% edge pruning, this is reduced to 110,000 MACs, a 64% reduction in message-passing computation. Similarly, for the GAT model on Cora, the baseline requires 2,865,000 MACs, which drops to 1,040,000 MACs (a 64% reduction) at 80% On PubMed with GCN, GNNShap reduces MACs from 2,058,000 to 711,000 (at 80% sparsity). For PubMed with GAT, the baseline requires 13,003 MACs, which is reduced to 4,493 MACs at 80% sparsity (a 65% reduction in computation). On Coauthor-CS with GAT, GNNShap reduces MACs from 100,530 to 50,804 at 55% sparsity, resulting in a 49% reduction. These significant reductions in MACs highlight the ability of Shapley value-based sparsification to maintain high accuracy while substantially lowering the computational cost of message passing.

While there are no significant differences among the other explainers, overall, PGExplainer tends to perform the worst. GrapSVX is also based on Shapley value and generally provides high-quality explanations; however, its consideration of nodes as players requires score conversion by averaging the scores of two connected nodes, which limits its sparsification performance. Moreover, we do not have GraphSVX results for the PubMed and Coauthor-CS datasets, as GraphSVX was unable to generate all node explanations within the 10-hour time limit.

5.2 Experimental Results with GLT Methods

In this section, we compare Shapley value-based sparsification (GNNShap) with GLT methods. Figure 4 shows test accuracies of GLT methods and GNNShap. UGS only utilizes training nodes in the gradient computation and learning process. Since training nodes are a small subset of the data, UGS has gradient information on a limited number of edges. For instance, 140 out of 2708 nodes were used in the training on the Cora dataset. Therefore, its graph pruning performance will be limited as shown in Figure 4. WD-GLT includes edges not involved in the training data in its loss function. This improves its sparsification capability compared to UGS on the Cora and PubMed datasets. While Fast-GLT gives competitive results, especially on PubMed, it loses considerable accuracy for higher sparsification ratios. On the other hand, Shapley value based GNNShap achieves significantly higher sparsity with minimal loss in accuracy.

The results show us that Shapley value based GNN explanations are better suited for graph sparsification due to the following limitations of GLT methods: (i) they require model training for each pruning percentage, and (ii) they have limited sparsification capability because of the necessity of labeled data. However, explainability approaches can learn the importance of edges for predicted classes, which eliminates the need for labels and enables higher pruning percentages with minimal loss of accuracy. Moreover, once edge scores are computed, there is no need to recompute edge scores for each sparsity level, making finding the ideal sparsity level much more effortless. A downside of using graph explainability scores to prune graphs is that it cannot sparsify model weights. However, starting with a small model and gradually increasing the model size until reaching a good accuracy requires less effort than starting with a large model and finding the ideal sparsity level by training the model multiple times.

5.3 Ablation Study

In this section, we investigate the effect of positive and negative attribution scores compared to non-negative explanation scores. For the non-negative GNNShap, we take the absolute value of GNNShap's scores and then compare the test accuracies. Figure 5 shows a significant decrease in GNNShap's pruning effectiveness

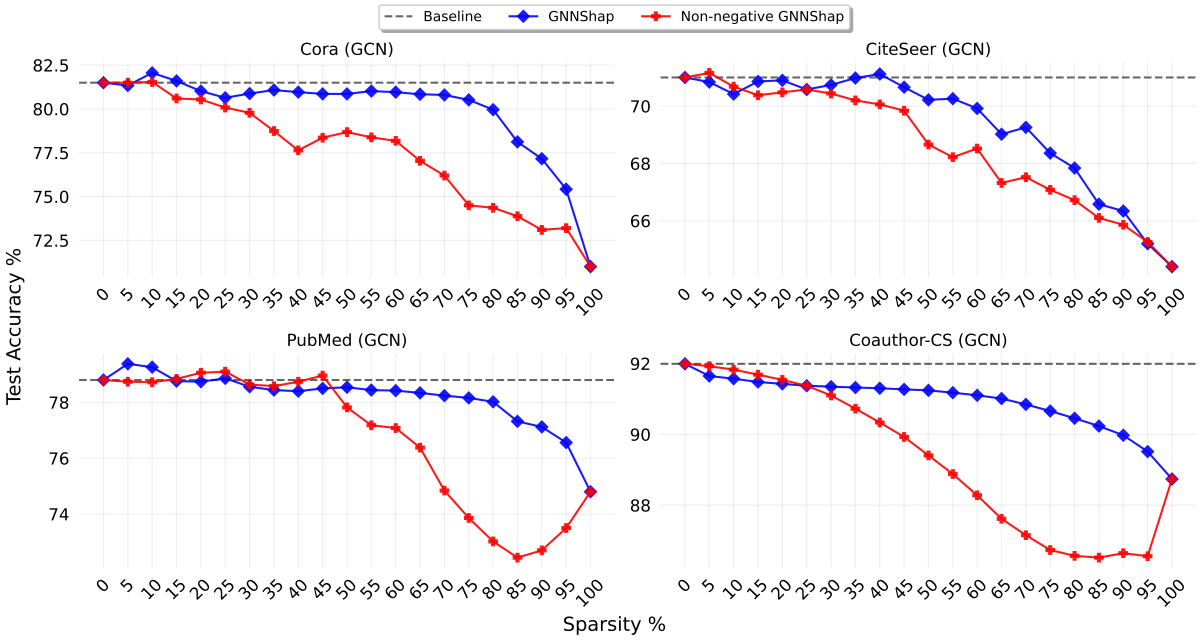

Figure 5: GNNShap test accuracy comparison when non-negative explanation scores are used. Non-negative scores significantly reduce test accuracy.

when non-negative scores are utilized for pruning. Considering negatively attributed edges as important (and thus not pruning them) introduces noise to the sparsified graph and reduces the pruning capability of GNNShap.

6 CONCLUSION

In this work, we have investigated the usability of Shapley values for graph sparsification. Shapley values provide both positive and negative explanation scores. This scoring mechanism enables more effective graph sparsification, thereby enhancing the efficiency and scalability of GNNs without compromising accuracy.

Our extensive evaluation demonstrates that Shapley value based sparsification achieves superior accuracy for more significant sparsification percentages, outperforming existing methods on three out of four datasets and across two models. Additionally, Shapley value based sparsification shows better sparsification ratios than graph lottery ticket approaches, highlighting its efficiency in reducing graph complexity.

However, a limitation of using explanation scores in sparsification is that if the underlying model does not perform well, the explanations generated can be misleading, as they are based on incorrect predictions. This limitation affects the applicability of Shapley values, as the reliability of explanations depends on the accuracy of the model.

In conclusion, Shapley value based graph sparsification successfully identifies important edges and provides more effective sparsification while maintaining the accuracy of GNNs. Future work can be designing a more effective aggregation scheme to combine local explanation scores with global importance scores. The improved aggregation scheme can enhance the reliability and applicability of Shapley values, resulting in even higher sparsities without compromising accuracy.

7 ACKNOWLEDGEMENTS

This research is partially supported by the Applied Mathematics Program of the DOE Office of Advanced Scientific Computing Research under contracts numbered DE-SC0022098 and DE-SC0023349 and by NSF grants CCF-2316234 and OAC-2339607.

REFERENCES

[1] Akkas, S., Azad, A., 2024. Gnnshap: Scalable and accurate gnn explanation using shapley values, in: Proceedings of the ACM Web Conference 2024, Association for Computing Machinery, New York, NY, USA. p. 827–838. URL: https://doi.org/10.1145/3589334.3645599, doi:10.1145/3589334.3645599.

[2] Baldassarre, F., Azizpour, H., 2019. Explainability techniques for graph convolutional networks. arXiv e-prints URL: https://arxiv.org/abs/1905.13686, arXiv:1905.13686. presented at the ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data.

[3] Chen, T., Sui, Y., Chen, X., Zhang, A., Wang, Z., 2021. A unified lottery ticket hypothesis for graph neural networks, in: Meila, M., Zhang, T. (Eds.), Proceedings of the 38th International Conference on Machine Learning, PMLR. pp. 1695–1706.

[4] Cheng, D., Zou, Y., Xiang, S., Jiang, C., 2025. Graph neural networks for financial fraud detection: a review. Frontiers of Computer Science 19, 1–15.

[5] Duval, A., Malliaros, F.D., 2021. Graphsvx: Shapley value explanations for graph neural networks, in: Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II 21, Springer. pp. 302–318.

[6] Gao, C., Wang, X., He, X., Li, Y., 2022. Graph neural networks for recommender system, in: Proceedings of the fifteenth ACM international conference on web search and data mining, pp. 1623–1625.

[7] Hui, B., Yan, D., Ma, X., Ku, W.S., 2023. Rethinking graph lottery tickets: Graph sparsity matters, in: The Eleventh International Conference on Learning Representations. URL: https://openreview.net/forum?id=fjh7UGQgOB.

[8] Khemani, B., Patil, S., Kotecha, K., Tanwar, S., 2024. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. Journal of Big Data 11, 18.

[9] Kipf, T.N., Welling, M., 2017. Semi-supervised classification with graph convolutional networks, in: International Conference on Learning Representations (ICLR).
[10] Li, G., Duda, M., Zhang, X., Koutra, D., Yan, Y., 2023a. Interpretable sparsification of brain graphs: Better practices and effective designs for graph neural networks, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 1223–1234.
[11] Li, X., Sun, L., Ling, M., Peng, Y., 2023b. A survey of graph neural network based recommendation in social networks. Neurocomputing 549, 126441.
[12] Liu, C., Ma, X., Zhan, Y., Ding, L., Tao, D., Du, B., Hu, W., Mandic, D.P., 2024. Comprehensive graph gradual pruning for sparse training in graph neural networks. IEEE Transactions on Neural Networks and Learning Systems 35, 14903–14917. doi:10.1109/TNNLS.2023.3282049.
[13] Luca, V.M.D., Longa, A., Lio, P., Passerini, A., 2024. xAI-drop: Don't use what you cannot explain, in: The Third Learning on Graphs Conference. URL: https://openreview.net/forum?id=adlpuqQD8Q.
[14] Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., Zhang, X., 2020. Parameterized explainer for graph neural network. Advances in neural information processing systems 33, 19620–19631.
[15] Luo, D., Cheng, W., Yu, W., Zong, B., Ni, J., Chen, H., Zhang, X., 2021. Learning to drop: Robust graph neural network via topological denoising, Association for Computing Machinery, New York, NY, USA. p. 779–787. URL: https://doi.org/10.1145/3437963.3441734, doi:10.1145/3437963.3441734.
[16] Ma, X., Ma, X., Erfani, S., Bailey, J., 2024. Training sparse graph neural networks via pruning and sprouting, in: Proceedings of the 2024 SIAM International Conference on Data Mining (SDM), SIAM. pp. 136–144.
[17] Mastropietro, A., Pasculli, G., Feldmann, C., Rodríguez-Pérez, R., Bajorath, J., 2022. Edgeshaper: Bond-centric shapley value-based explanation method for graph neural networks. Iscience 25.
[18] Min, S., Gao, Z., Peng, J., Wang, L., Qin, K., Fang, B., 2021. Stgsn—a spatial–temporal graph neural network framework for time-evolving social networks. Knowledge-Based Systems 214, 106746.
[19] Motie, S., Raahemi, B., 2024. Financial fraud detection using graph neural networks: A systematic review. Expert Systems with Applications 240, 122156.
[20] Naik, H.G., Polster, J., Shekhar, R., Horváth, T., Turán, G., 2024. Iterative graph neural network enhancement via frequent subgraph mining of explanations. arXiv preprint arXiv:2403.07849 .
[21] Pereira, T., Nascimento, E., Resck, L.E., Mesquita, D., Souza, A., 2023. Distill n'explain: explaining graph neural networks using simple surrogates, in: International Conference on Artificial Intelligence and Statistics, PMLR. pp. 6199–6214.
[22] Perotti, A., Bajardi, P., Bonchi, F., Panisson, A., 2023. Explaining identity-aware graph classifiers through the language of motifs, in: 2023 International Joint Conference on Neural Networks (IJCNN), IEEE. pp. 1–8.
[23] Pope, P.E., Kolouri, S., Rostami, M., Martin, C.E., Hoffmann, H., 2019. Explainability methods for graph convolutional neural networks, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10772–10781.
[24] Shapley, L.S., 1951. Notes on the N-Person Game – II: The Value of an N-Person Game. RAND Corporation. doi:10.7249/RM0670.
[25] Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S., 2018. Pitfalls of graph neural network evaluation. arXiv e-prints arXiv:1811.05868. presented at the Relational Representation Learning Workshop (R2L 2018), NeurIPS 2018.
[26] Shin, Y.M., Shin, W.Y., 2024. On the feasibility of fidelity for graph pruning, in: IJCAI Workshop on Explainable AI (XAI). URL: https://arxiv.org/abs/2406.11504.
[27] Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M., 2015. Striving for simplicity: The all convolutional net, in: International Conference on Learning Representations (ICLR) Workshop Track. URL: https://arxiv.org/abs/1412.6806.
[28] Sui, Y.D., Wang, X., Chen, T., Wang, M., He, X.N., Chua, T.S., 2024. Inductive lottery ticket learning for graph neural networks. Journal of Computer Science and Technology 39, 1223–1237.
[29] Sundararajan, M., Taly, A., Yan, Q., 2017. Axiomatic attribution for deep networks, in: International conference on machine learning, PMLR. pp. 3319–3328.
[30] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks, in: International Conference on Learning Representations. URL: https://openreview.net/forum?id=rJXMpikCZ.
[31] Villani, C., Villani, C., 2009. The wasserstein distances. Optimal transport: old and new , 93–111.
[32] Vu, M., Thai, M.T., 2020. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. Advances in neural information processing systems 33, 12225–12235.
[33] Wang, B., Chen, J., Li, C., Zhou, S., Shi, Q., Gao, Y., Feng, Y., Chen, C., Wang, C., 2024. Distributionally robust graph-based recommendation system, in: Proceedings of the ACM Web Conference 2024, pp. 3777–3788.
[34] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K., 2019. Simplifying graph convolutional networks, in: International conference on machine learning, Pmlr. pp. 6861–6871.
[35] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S., 2020. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems 32, 4–24.
[36] Yang, Z., Cohen, W., Salakhudinov, R., 2016. Revisiting semi-supervised learning with graph embeddings, in: International conference on machine learning, PMLR. pp. 40–48.
[37] Ying, Z., Bourgeois, D., You, J., Zitnik, M., Leskovec, J., 2019. Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems 32.
[38] You, H., Lu, Z., Zhou, Z., Fu, Y., Lin, Y., 2022. Early-bird gcns: Graph-network co-optimization towards more efficient gcn training and inference via drawing early-bird lottery tickets, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 8910–8918.
[39] Yuan, H., Yu, H., Gui, S., Ji, S., 2023. Explainability in graph neural networks: A taxonomic survey. IEEE Transactions on Pattern Analysis & Machine Intelligence 45, 5782–5799. doi:10.1109/TPAMI.2022.3204236.
[40] Yuan, H., Yu, H., Wang, J., Li, K., Ji, S., 2021. On explainability of graph neural networks via subgraph explorations, in: International conference on machine learning, PMLR. pp. 12241–12252.
[41] Yue, Y., Zhang, G., Yang, H., Cheng, D., 2025. Fast track to winning tickets: Repowering one-shot pruning for graph neural networks. Proceedings of the AAAI Conference on Artificial Intelligence 39, 13152–13160.
[42] Zhang, G., Sun, X., Yue, Y., Jiang, C., Wang, K., Chen, T., Pan, S., 2025. Graph sparsification via mixture of graphs, in: The Thirteenth International Conference on Learning Representations. URL: https://openreview.net/forum?id=7ANDviElAo.
[43] Zhang, G., Wang, K., Huang, W., Yue, Y., Wang, Y., Zimmermann, R., Zhou, A., Cheng, D., Zeng, J., Liang, Y., 2024a. Graph lottery ticket automated, in: The Twelfth International Conference on Learning Representations. URL: https://openreview.net/forum?id=nmBjBZoySX.
[44] Zhang, G., Yue, Y., Wang, K., Fang, J., Sui, Y., Wang, K., Liang, Y., Cheng, D., Pan, S., Chen, T., 2024b. Two heads are better than one: Boosting graph sparse training via semantic and topological awareness, in: Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., Berkenkamp, F. (Eds.), Proceedings of the 41st International Conference on Machine Learning, PMLR. pp. 60197–60219. URL: https://proceedings.mlr.press/v235/zhang24bx.html.
[45] Zheng, C., Zong, B., Cheng, W., Song, D., Ni, J., Yu, W., Chen, H., Wang, W., 2020. Robust graph representation learning via neural sparsification, in: III, H.D., Singh, A. (Eds.), Proceedings of the 37th International Conference on Machine Learning, PMLR. pp. 11458–11468.
[46] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI open 1, 57–81.

A MORE AGGREGATION METHODS

We provide two alternative aggregation results: **sum** and **weighted mean**. While sum uses the sum of scores as a global mask, the weighted mean uses model prediction as weights and applies the weighted mean. Figure 6 and 7 show these aggregation results. However, we don't see a significant difference compared to mean aggregation.

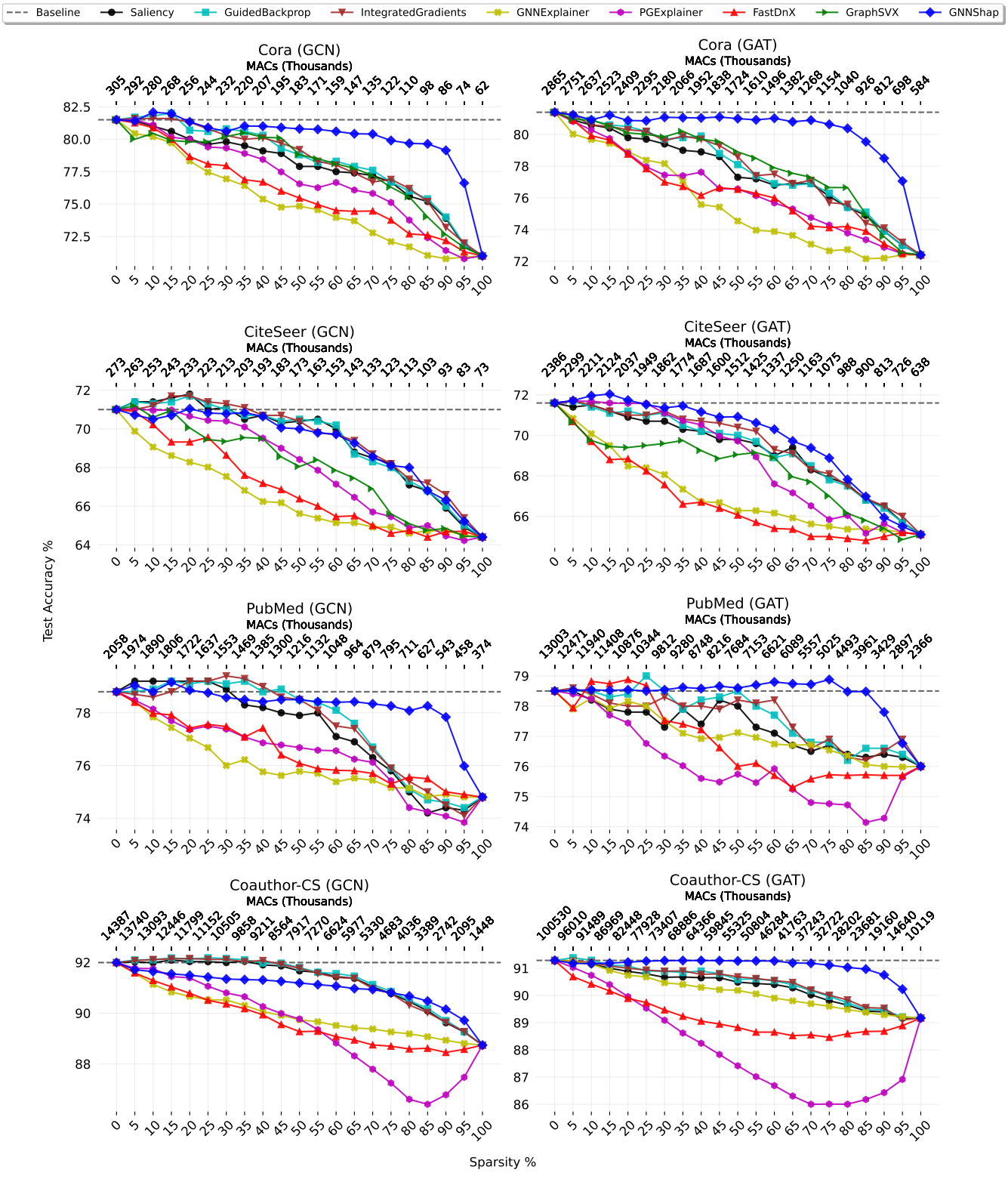

Figure 6: Test accuracies when edges sparsified using sum aggregated explanation scores. GNNShap gives competitive or even better accuracies for high sparsification percentages.

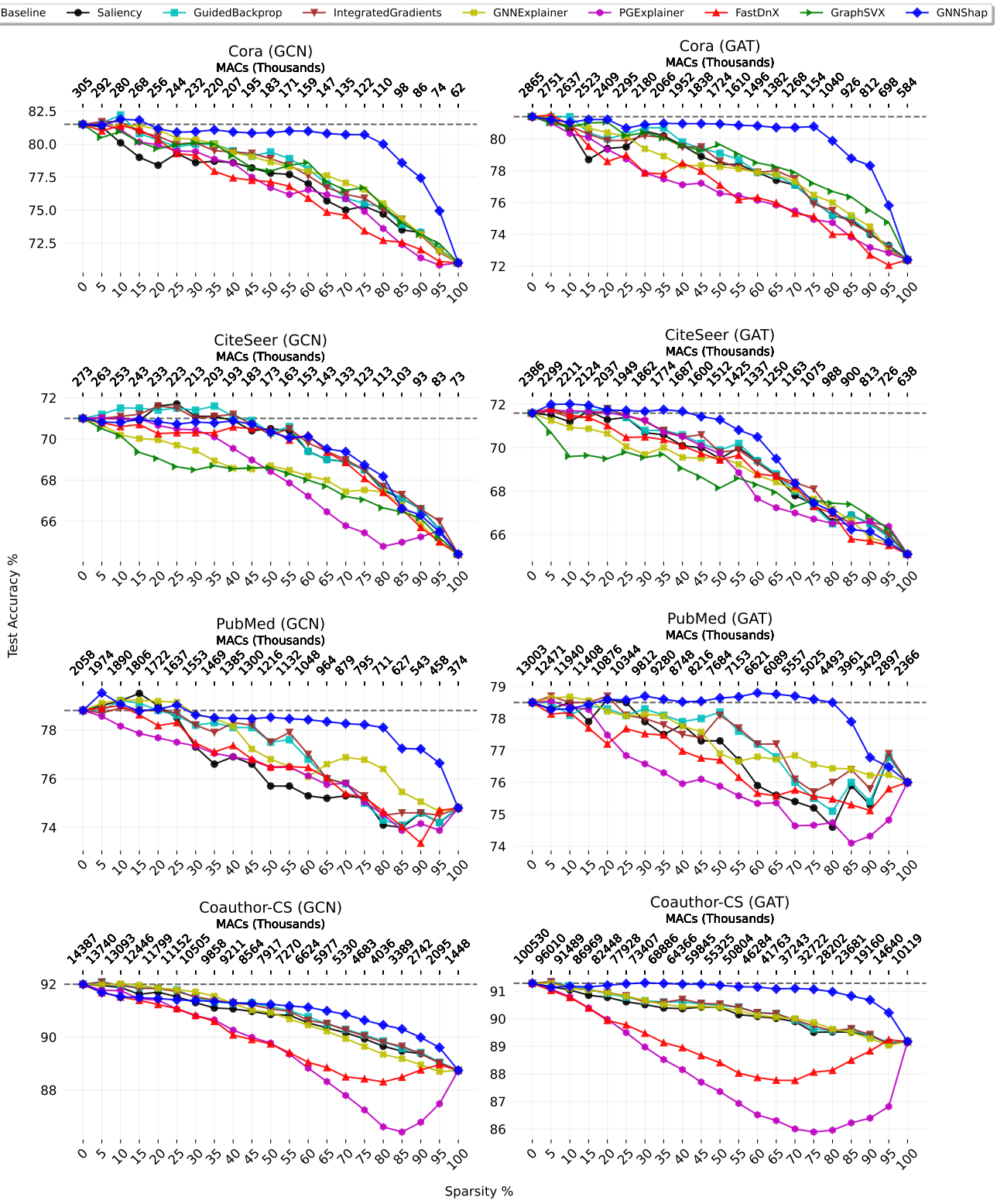

Figure 7: Test accuracies when edges sparsified using weighted mean aggregated explanation scores. GNNShap gives competitive or even better accuracies for high sparsification percentages.