Sanity Checks for Evaluating Graph Unlearning

Varshita Kolipaka * **Akshit Sinha** * **Debangan Mishra** **Sumit Kumar** **Arvindh Arun**
Shashwat Goel **Ponnurangam Kumaraguru**
IIIT Hyderabad, India
varshita.k@research.iiit.ac.in, akshit.sinha@students.iiit.ac.in

Abstract

Graph neural networks (GNNs) are increasingly being used on sensitive graph-structured data, necessitating techniques for handling unlearning requests on the trained models, particularly node unlearning. However, unlearning nodes on GNNs is challenging due to the interdependence between the nodes in a graph. We compare MEGU, a state-of-the-art graph unlearning method, and SCRUB, a general unlearning method for classification, to investigate the efficacy of graph unlearning methods over traditional unlearning methods. Surprisingly, we find that SCRUB performs comparably or better than MEGU on random node removal and on removing an adversarial node injection attack. Our results suggest that 1) graph unlearning studies should incorporate general unlearning methods like SCRUB as baselines, and 2) there is a need for more rigorous behavioral evaluations that reveal the differential advantages of proposed graph unlearning methods. Our work, therefore, motivates future research into more comprehensive evaluations for assessing the true utility of graph unlearning algorithms.

1 Introduction

Graph Neural Networks (GNNs) have found widespread application across various domains, ranging from biology to weather forecasting (Kipf & Welling, 2017; Perozzi et al., 2014; Veličković et al., 2018), primarily for tasks involving node classification on graph-structured data. These models often leverage sensitive data during training, such as personal details about individuals used in recommendation systems (Friedman et al., 2015). Requests to “remove” information from the model were a primary motivation for unlearning (Bourtoule et al., 2021). However, unlearning is also motivated by the need to address poisoning attacks, noisy labels, and outdated information, where retraining from scratch is expensive (Goel et al., 2024). In graphs, unlearning requests typically take the form of node, feature, or edge unlearning (Cheng et al., 2023). Node unlearning involves removing the influence of a specific set of nodes from a model, which is challenging due to the interdependence between nodes resulting from the graph structure and the message-passing mechanisms commonly employed by GNNs.

Recent studies have proposed graph unlearning methods specifically designed for node unlearning, where removing one node can impact the representations of its neighboring nodes (Said et al., 2023). A common evaluation setup involves randomly deleting a subset of nodes, modeling user privacy requests (Qu et al., 2023). However, the development of new graph-specific unlearning algorithms is warranted only if standard unlearning techniques (such as those from fields like image classification) perform worse than graph methods on these tasks.

In our review of the existing graph unlearning literature, we find that papers do not comprehensively validate the assumptions regarding the need for graph-specific unlearning methods and evaluation setups. We compare a state-of-the-art graph unlearning method, MEGU (Li et al., 2024), with SCRUB (Kurmanji et al., 2024), a recently proposed technique for general classification unlearning tasks. Surprisingly, we find that SCRUB matches or outperforms MEGU’s performance on the random node deletion setup commonly used in the graph unlearning literature (Li et al., 2024; Cheng et al., 2023). As a first step towards stronger evaluations designed to highlight interdependencies, we propose testing unlearning to remove the adverse effects of a node injection poisoning attack. Our preliminary investigation sheds light on two main points:

1. **Incorporate methods from other areas**, such as SCRUB from image classification, as baseline comparisons. Even though they are not tailored for graph-specific properties, these methods perform surprisingly well in current evaluations.
2. **Need for more rigorous evaluations of graph unlearning,** that isolate graph-specific properties and show the differential effectiveness of unlearning methods targeted toward graph tasks.

* Equal contribution

2 Problem Setting

We perform a semi-supervised multi-label classification using a Graph Convolutional Network (GCN) (Kipf & Welling, 2017) and compute loss using cross-validation. Node unlearning in graph neural networks involves removing the influence of specific nodes, which may have manipulated features or perturb the topology of the original graph. We perform node-level unlearning in two settings: the random node removal (a benign form of unlearning request common in graph unlearning evaluations) and node injection attack removal, an adversarial form of unlearning requests.

2.1 Desiderata

The overarching objective for all unlearning requests is to forget efficiently, while minimising the impact on the overall model utility (Kurmanji et al., 2024). Ideally, the performance of the unlearned GCN should resemble that of a fresh GCN retrained solely on clean samples. We aim to move away from the current forget set labels without assuming ground truth label knowledge.

In graph unlearning literature, various proxies measure "forgetting" - Accuracy (Guo et al., 2019), F1 scores (Cong & Mahdavi, 2023; Wu et al., 2023), and Membership Inference Attacks (Yeom et al., 2018; Cheng et al., 2023; Olatunji et al., 2021) (commonly used in privacy settings). Our evaluation considers two settings: random deletion requests and poisoned node deletion. We compute Micro-F1 score, which is the main metric used to report MEGU's results (Li et al., 2024), as well as Accuracy and Poison Success Rate (Lin et al., 2021).

2.2 Formulation and Notation

Graph Notation. We formalize node unlearning similar to Li et al. (2024),

- $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$ represents a graph $\mathcal{G}$ with $|\mathcal{V}| = n$ nodes, $|\mathcal{E}| = m$ edges, and $\mathcal{X}(i)$ corresponding to the feature vector of node i, where V_{tr} corresponds to the set of nodes used for training and V_{test} corresponds to the set of nodes used for testing.
- *Forget Set*: Let $\mathcal{V}_f \subseteq \mathcal{V}_{tr}$ denote the set of nodes to be unlearnt.
- *Retain Set*: Let $\mathcal{V}_r \subseteq \mathcal{V}_{tr}$ denote the set of nodes to be retained.

Model Utility. We operationalise this by computing the Micro-F1 score on $\mathcal{V}_{test}$. An ideal unlearning method should minimally impair the performance of the model, as tested against clean data points.

Influence of Samples to be Removed. We operationalize this as the Micro-F1 score on samples that are in the forget set ($\mathcal{V}_f \subseteq \mathcal{V}_{tr}$). An effective unlearning method aims to minimise this score. In case of unlearning injection attacks, however, we measure this by applying the poison to a copy of the test nodes, where it is also referred to as *Poison Success Rate.*

2.3 Methods Tested

SCalable Remembering and Unlearning unBound (SCRUB) (Kurmanji et al., 2024) is a relatively new general unlearning method proposed for a variety of tasks, such as preserving user privacy, resolving confusion, and removing biases. SCRUB employs a teacher-student framework to handle unlearning requests. It is not bound by any assumptions or constrained definitions of unlearning, such as closeness to the retrained model in the state space. SCRUB highlights the need to consider diverse applications, requirements, and evaluations for proposed unlearning methods, which partially motivates us to test it on graph unlearning tasks. Despite being a state-of-the-art unlearning method in the image classification literature, it has not been used as a baseline in the graph unlearning literature.

Mutual Evolution Graph Unlearning (MEGU) (Li et al., 2024), the current SOTA for graph unlearning, has a mutually trained Linear Unlearning Module which aims to remove the influence of $\mathcal{V}_f$, and a Predictive Module which aims to retain the performance on $\mathcal{V}_r$. It uses a mutual evolution approach relying on the forgetting capability of the unlearning module, and the reasoning capability of the predictive module. Unlike SCRUB, MEGU leverages graph topology by identifying the Highly Influenced Nodes (HINs) to act as anchors in its loss formulation and employs a topology-aware unlearning propagation using homophily and feature smoothing.

3 Experiments

3.1 Setup

We evaluate on two benchmark datasets, Cora and CiteSeer (Bojchevski & Günnemann, 2017), commonly used in graph unlearning literature. Cora contains 2,708 nodes, each with 1,433 features belonging to one of its 7 classes and 5,429 unique edges. CiteSeer contains 3,327 nodes, each with 3,703 features belonging to one of its 6 classes and 4,732 unique edges.

We split both datasets following the guidelines of recent graph unlearning approaches (Cheng et al., 2023; Li et al., 2024), randomly assigning 80% of nodes for training and 20% for testing. For simplicity and generalizability, we limit the scope of our experiments to the GCN model. In all experiments, we train a 2-layer GCN with a hidden dimension of 16, using an Adam Optimizer with a learning rate of 0.025 and weight decay of 5×10^{-4}, and train for 200 epochs.

To ensure a fair comparison between the unlearning methods, both SCRUB and MEGU are run for 100 epochs. For each dataset, we also conduct an extensive search on the hyperparameters recommended in the respective papers, as described in Appendix A.

3.2 Evaluation 1 - Random Unlearning

Motivation. Unlearning random nodes is the prevalent evaluation to benchmark Graph Unlearning works (Said et al., 2023). This form of unlearning request is typical in privacy related scenarios. In this setting, a subset of nodes belonging to $\mathcal{V}_{tr}$ are randomly chosen to be deleted from the graph. We vary the size of the deletion set from 5% to 50% of the total nodes in the training set.

Table 1: ***Random node unlearning results on Cora and CiteSeer.*** $F1_{\mathcal{V}_{\text{test}}}$ indicates F1 score on the V_{test} where higher indicates more model utility. $F1_{\mathcal{V}_f}$ indicates the F1 score on $\mathcal{V}_f$, where lower indicates more forgetting.

Deletion Size	**Retrain From Scratch**		**SCRUB**		**MEGU**	
	$F1_{\mathcal{V}_{test}}$	$F1_{\mathcal{V}_f}$	$F1_{\mathcal{V}_{test}}$	$F1_{\mathcal{V}_f}$	$F1_{\mathcal{V}_{test}}$	$F1_{\mathcal{V}_f}$
Cora						
5%	86.24%	83.33%	85.87%	87.50%	85.87%	87.50%
10%	86.61%	84.97%	85.87%	87.56%	85.69%	87.56%
20%	85.32%	86.53%	86.06%	89.64%	85.87%	89.64%
30%	85.14%	87.22%	86.06%	89.81%	85.87%	87.50%
40%	86.24%	83.33%	85.87%	87.50%	85.87%	90.80%
50%	85.69%	86.74%	86.24%	90.36%	85.87%	90.36%
CiteSeer						
5%	81.91%	72.36%	80.57%	75.61%	80.12%	75.61%
10%	81.32%	74.09%	81.02%	76.11%	80.27%	76.11%
20%	83.11%	72.93%	80.57%	75.56%	80.42%	75.56%
30%	82.36%	73.22%	81.02%	75.64%	80.42%	75.64%
40%	81.91%	74.27%	81.02%	76.59%	80.42%	76.19%
50%	82.06%	74.01%	80.57%	76.92%	79.97%	76.76%

Results. Random node unlearning results are presented in Table 1. SCRUB performs on par with MEGU on both retaining model utility and lowering accuracy on the forget set across deletion sizes. Both methods are worse than the retrain-from-scratch gold standard. These results suggest that general unlearning methods may perform just as well as specialised graph unlearning methods under this evaluation.

3.3 Evaluation 2 - Unlearning Injection Attacks

Motivation. The interconnected nature of graph data introduces a critical vulnerability: a small number of manipulated or poisoned nodes in the graph can adversely affect the performance across the entire network. These manipulations can arise from various sources, such as adversarial attacks, data biases, or even errors in the feature or label collection process (Goel et al., 2024). Consequently, there may be scenarios where we identify these manipulated nodes after training a GNN model and wish to remove their influence without retraining the entire model from scratch.

Adversary's Perspective. From the adversary's point of view, they can inject an arbitrary number of nodes with a feature vector of their choice into the graph. This simulates scenarios that necessitate unlearning requests, such as removing backdoors, biases, and confusions, manifesting as spurious correlations learned due to mislabeling or feature errors.

Poisoning Attack. For *Cora*, we inject 50 poisoned nodes with a trigger size of 50 features, targeting Class 1 as the poisoned label. For *CiteSeer*, we inject 30 poisoned nodes with a trigger size of 150 features, targeting Class 0 as the poisoned label. The target classes are chosen to be consistent with MEGU's evaluation setting.

Table 2: Poisoned node unlearning results on Cora and CiteSeer. SCRUB as proficient as MEGU at retaining model utility, and even outperforms MEGU in deleting the effects of the poison as measured by the decrease in the poison success rate. F1 Score results, which indicate model utility, are presented on the clean test set of the data. (↑) indicates a higher value is better, while (↓) indicates a lower value is better. Best performance is indicated in **bold**.

	Cora		CiteSeer	
Unlearning Method	**F1 Score** (↑)	**Poison Success Rate** (↓)	**F1 Score** (↑)	**Poison Success Rate** (↓)
No Unlearning	90.83%	98.53%	**80.41%**	99.70%
SCRUB	91.19%	**0%**	78.62%	**0%**
MEGU	**93.39%**	9.17%	76.53%	11.95%

Metrics. We evaluate the unlearning methods using the following metrics:

1. **Retention of Model Utility (F1)** To assess the model's overall performance after unlearning, we measure the F1 score on the original test set $\mathcal{V}_{test}$. The F1 score, which aggregates the contributions of all classes, is particularly useful for evaluating multi-class accuracy, especially in imbalanced datasets.
2. **Poison Success Rate (PSR)** The proportion of times the model predicts the adversary's intended labels (poison labels) in the poisoned dataset, or formally, $F1_{\mathcal{V}_f}$. A higher PSR (closer to 1) indicates a more successful poison attack. On the other hand, the unlearning method's ability to mitigate the adverse effects of the poisoned nodes is measured by the decrease in PSR after unlearning.

Results. The results of the evaluation are presented in Table 2. We observe that GCN is highly susceptible to these attacks, as the classical poisoning attack achieves a high poison trigger success rate while maintaining the overall model utility. We find that not only is SCRUB as proficient as MEGU at retaining model utility, it even **outperforms MEGU in deleting the effects of the poison.** These results suggest that general unlearning methods are an important baseline even for graph unlearning settings. Further, given that both evaluations fail to differentiate SCRUB and MEGU, there is a need for evaluations that better measure the unlearning of propagated poisoning due to message passing in GNNs.

4 Conclusion

In this work, we compare a state-of-the-art method proposed specifically for graph unlearning (MEGU) with a generic classification unlearning method (SCRUB) and note that their performances are comparable in the commonly tested setting of random node deletions. We hypothesize that this may be because the random node deletion evaluation is not strong enough to test the interesting aspect of graph unlearning, i.e., the interdependence of node representations. We then test the unlearning using a simple node injection poisoning attack and find that on the same metrics used to report MEGU's performance, SCRUB outperforms on this task. Thus, our work shows the importance of including standard unlearning baselines in the graph unlearning setting, as they may be surprisingly effective. It also follows that there is a need to design rigorous evaluations for graph unlearning.

References

Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

Lucas Bourtoule, Varun Chandrasekaran, Christopher Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *Proceedings of the 42nd IEEE Symposium on Security and Privacy*, 2021.

Jiali Cheng, George Dasoulas, Huan He, Chirag Agarwal, and Marinka Zitnik. GNNDelete: A general unlearning strategy for graph neural networks. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=X9yCkmT5Qrl.

Weilin Cong and Mehrdad Mahdavi. Efficiently forgetting what you have learned in graph representation learning via projection. In *International Conference on Artificial Intelligence and Statistics*, pp. 6674–6703. PMLR, 2023.

Arik Friedman, Bart Knijnenburg, Kris Vanhecke, Luc Martens, and Shlomo Berkovsky. *Privacy Aspects of Recommender Systems*, pp. 649–688. 01 2015. ISBN 978-1-4899-7636-9. doi: 10.1007/978-1-4899-7637-6_19.

Shashwat Goel, Ameya Prabhu, Philip Torr, Ponnurangam Kumaraguru, and Amartya Sanyal. Corrective machine unlearning. *arXiv preprint arXiv:2402.14015*, 2024.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded machine unlearning. *Advances in Neural Information Processing Systems*, 36, 2024.

Xunkai Li, Yulin Zhao, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, and Guoren Wang. Towards effective and general graph unlearning via mutual evolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13682–13690, 2024.

Jing Lin, Long Dang, Mohamed Rahouti, and Kaiqi Xiong. Ml attack models: Adversarial attacks and data poisoning attacks. *arXiv preprint arXiv:2112.02797*, 2021.

Iyiola E Olatunji, Wolfgang Nejdl, and Megha Khosla. Membership inference attack on graph neural networks. In *2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pp. 11–20. IEEE, 2021.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '14. ACM, August 2014. doi: 10.1145/2623330.2623732. URL http://dx.doi.org/10.1145/2623330.2623732.

Youyang Qu, Xin Yuan, Ming Ding, Wei Ni, Thierry Rakotoarivelo, and David Smith. Learn to unlearn: A survey on machine unlearning. *arXiv preprint arXiv:2305.07512*, 2023.

Anwar Said, Tyler Derr, Mudassir Shabbir, Waseem Abbas, and Xenofon Koutsoukos. A survey of graph unlearning. *arXiv preprint arXiv:2310.02164*, 2023.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

Jiancan Wu, Yi Yang, Yuchun Qian, Yongduo Sui, Xiang Wang, and Xiangnan He. Gif: A general graph unlearning strategy via influence function. In *Proceedings of the ACM Web Conference 2023*, pp. 651–661, 2023.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pp. 268–282. IEEE, 2018.

A HYPERPARAMETER TUNING

This section contains details about the hyperparameter tuning performed for both MEGU and SCRUB. We follow the authors and use the same set of hyperparameters for evaluations.

Table 3: Candidate values used for hyperparameter tuning

Hyperparameter	Values
SCRUB	
α	[0.5, 1, 2]
$msteps$	[10, 25, 50]
Learning Rate	[0.025, 0.01, 0.001]
MEGU	
$\alpha 1$	[0.1, 0.5, 0.8, 0.24]
$\alpha 2$	[0.1, 0.5, 0.8, 0.12]
κ	[0.01, 0.1, 0.05]
Learning Rate	[0.01, 0.05, 0.09, 0.1]

Table 4: Selected hyperparameters for hyperparameter tuning for SCRUB and MEGU over Cora and CiteSeer

Dataset	SCRUB			MEGU			
	α	$msteps$	Learning Rate	$\alpha 1$	$\alpha 1$	κ	Learning Rate
Cora	1	10	0.01	0.8	0.5	0.01	0.09
CiteSeer	0.05	25	0.01	0.8	0.5	0.1	0.1