Outline

The appendix is organized as follows:

A.1 Dataset Description

The 9 statistics of graph benchmark datasets and feature, node, edge-level unlearning requests are shown in Table.6. Moreover, the description of all datasets are listed below:

Cora, **CiteSeer**, and **PubMed** (Yang, Cohen, and Salakhutdinov 2016) are three citation network datasets representing undirected graphs, where nodes represent papers and edges represent citation relationships between papers. The node features are word vectors, where each element is a binary variable (0 or 1) indicating the presence or absence of each word in the paper.

Coauthor CS and **Coauthor Physics** (Shchur et al. 2018) are co-authorship graphs based on the Microsoft Academic Graph. Here, nodes are authors, that are connected by an edge if they co-authored a paper; node features represent paper keywords for each author's papers, and class labels indicate the most active fields of study for each author.

Amazon Photo and **Amazon Computers** (Shchur et al. 2018) are segments of the Amazon co-purchase graph. Nodes represent goods and edges represent that two goods are frequently bought together. Given product reviews as bag-of-words node features, the task is to map goods to their respective product category.

PPI (Zeng et al. 2020) stands for Protein-Protein Interaction (PPI) network, where nodes represent protein. If two proteins participate in a life process or perform a certain function together, it is regarded as an interaction between these two proteins. Complex interactions between multiple proteins can be described by PPI networks.

Flickr (Zeng et al. 2020) dataset originates from the SNAP, they collect Flickr data and generate an undirected graph. Nodes represent images, and edges connect images with common properties like geographic location, gallery, or shared comments. Node features are 500-dimensional bag-of-words representations extracted from the images. The labels are manually merged from the 81 tags into 7 classes.

A.2 Compared Baselines

Backbone GNNs. To evaluate the effectiveness of various GU strategies, we have selected commonly used GNNs as the backbone models to simulate scenarios where unlearning requests are received during training. The chosen models encompass GCN (Kipf and Welling 2017), GAT (Veličković et al. 2018), GraphSage (Hamilton, Ying, and Leskovec 2017), GIN (Xu et al. 2019), SGC (Wu et al. 2019), Cluster-GCN (Chiang et al. 2019), and GraphSAINT (Zeng et al. 2020). These models represent successful recent designs in graph learning, widely applicable in both transductive and inductive settings. Furthermore, various backbone GNNs can be employed to assess the generalization capabilities of diverse GU approaches. The salient characteristics of all baseline models are outlined below:

GCN (Kipf and Welling 2017) introduces a novel approach to graph-structured data that uses an efficient layer-wise propagation rule that is based on a first-order approximation of spectral convolutions on graphs.

GAT (Veličković et al. 2018) utilizes a graph attention layer to assign varying importance to different nodes within a neighborhood, thus better-representing graph information.

GIN (Xu et al. 2019) develops a simple graph learning architecture with MLP that is as powerful as the Weisfeiler-Lehman graph isomorphism test.

SGC (Wu et al. 2019) simplifies GCN by removing nonlinearities and collapsing weight matrices between consecutive layers, bringing higher running efficiency.

GraphSage (Hamilton, Ying, and Leskovec 2017) is an inductive framework that leverages neighbor node attribute information to efficiently generate representations.

Cluster-GCN (Chiang et al. 2019) is a novel GNN designed for training with Stochastic Gradient Descent (SGD) by leveraging the graph clustering structure.

GraphSAINT (Zeng et al. 2020) is a novel inductive learning method that enhances training efficiency and accuracy through graph sampling.

Graph Unlearning strategies. In our experimental study, we delineate the characteristics and provide descriptions of GU strategies that have been proposed in recent years:

GraphEraser (Chen et al. 2022) propose a novel machine unlearning framework tailored to graph data. Its contributions include two novel graph partition algorithms and a learning-based aggregation method.

GUIDE (Wang, Huai, and Wang 2023) improves GraphEraser by the graph partitioning with fairness and balance, efficient subgraph repair, and similarity-based aggregation.

CGU (Chien, Pan, and Milenkovic 2022) presents the underlying analysis of certified GU using SGC and their generalized PageRank (GPR) extensions as examples.

GIF (Wu et al. 2023) incorporates an additional loss term for influenced neighbors, considering structural dependencies, and provides a closed-form solution for better understanding the unlearning mechanism.

Projector (Cong and Mahdavi 2023) achieves unlearning by projecting the weights of the pre-trained linear model onto a subspace that is unrelated to the unlearning entities.

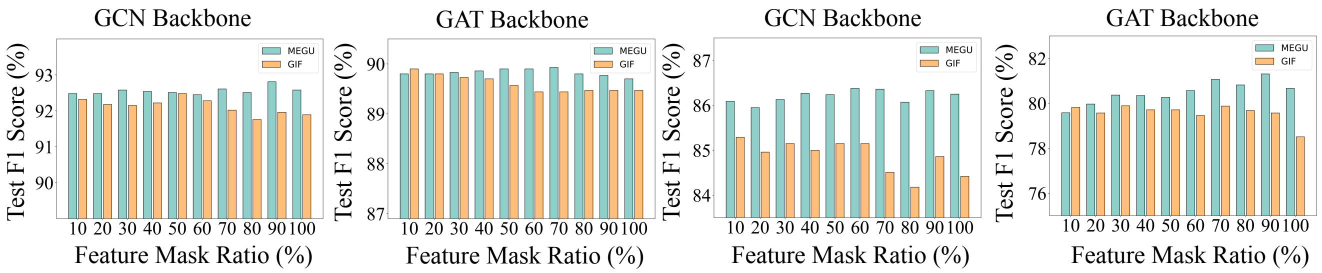
GNNDelete (Cheng et al. 2023) is a novel model-agnostic layer-wise operator designed to optimize topology influence in the graph unlearning requests.

A.3 Hyperparameter settings

The hyperparameters in the backbones and GU approaches are set according to the original paper if available. Otherwise, we perform an automatic hyperparameter search via the Optuna (Akiba et al. 2019). Specifically, we explore the optimal shards within the ranges of 20 to 100. The weight coefficients of the loss function and other hyperparameter is get by means of an interval search from $\{0, 1\}$ or the interval suggested in the original paper. For our proposed MEGU,

Table 6: The statistics of the experimental datasets.

Dataset	#Nodes	#Features	#Edges	#Classes	#Feat./Node/Edge Unlearn	Task type	Description
Cora	2,708	1,433	5,429	7	216/216/802	Transductive	citation network
CiteSeer	3,327	3,703	4,732	6	266/266/736	Transductive	citation network
PubMed	19,717	500	44,338	3	1,577/1,577/5,426	Transductive	citation network
Amazon Photo	7,487	745	119,043	8	612/612/5,889	Transductive	co-purchase graph
Amazon Computers	13,381	767	245,778	10	1100/1100/10651	Transductive	co-purchase graph
Coauthor CS	18,333	6,805	81,894	15	1,466/1,466/9,081	Transductive	co-authorship graph
Coauthor Physics	34,493	8,415	247,962	5	2,759/2,759/21,712	Transductive	co-authorship graph
PPI	56,944	50	818,716	121	4,555/4,555/39,993	Inductive	protein interactions network
Flickr	89,250	500	899,756	7	7,140/7,140/47,449	Inductive	image network



(a) Photo

(b) Computers

Figure 5: Performance of different feature mask ratios on Photo and Computers with GCN and GAT backbone.

Table 7: Detailed hyperparameter setting on all datasets.

Dataset	unlearning rate	κ	α_1	α_2
Cora	0.05	0.01	0.8	0.5
CiteSeer	0.09	0.01	0.24	0.12
PubMed	0.04	0.09	0.18	0.12
Amazon Photo	0.065	0.06	0.94	0.2
Amazon Computers	0.001	0.01	0.05	0.05
Coauthor CS	0.007	0.01	0.03	0.13
Coauthor Physics	0.04	0.1	0.02	0.27
PPI	0.03	0.08	-	-
Flickr	0.001	0.01	0.05	0.05

the coefficient of personalized PageRank in the context of the topology-aware unlearning propagation process (α) and loss function (κ) are explored within the ranges of 0 to 1. More details can be referred to Eq. (3) and Eq. (7).

To help reproduce the experimental results, we provide the hyperparameter settings in Table 7, where α_1 and α_2 correspond to the $\mathbf{E}_v$ and $\hat{\mathbf{Y}}_v + \mathbf{E}_v$ propagation coefficients in Eq. (3). The hyperparameters presented in this table are applicable to all backbones mentioned in this paper. Since the PPI dataset is multi-label classification task, in order to avoid propagating high bias on the graph due to multi-label classification, we did not use the Topo. UP module when processing this dataset, therefore the PPI dataset does not have the corresponding α_1 and α_2. In addition, we use SGD as the optimizer and set the number of epochs to 100. Specific experimental strategies and examples can be found in https://github.com/xkLi-Allen/MEGU.

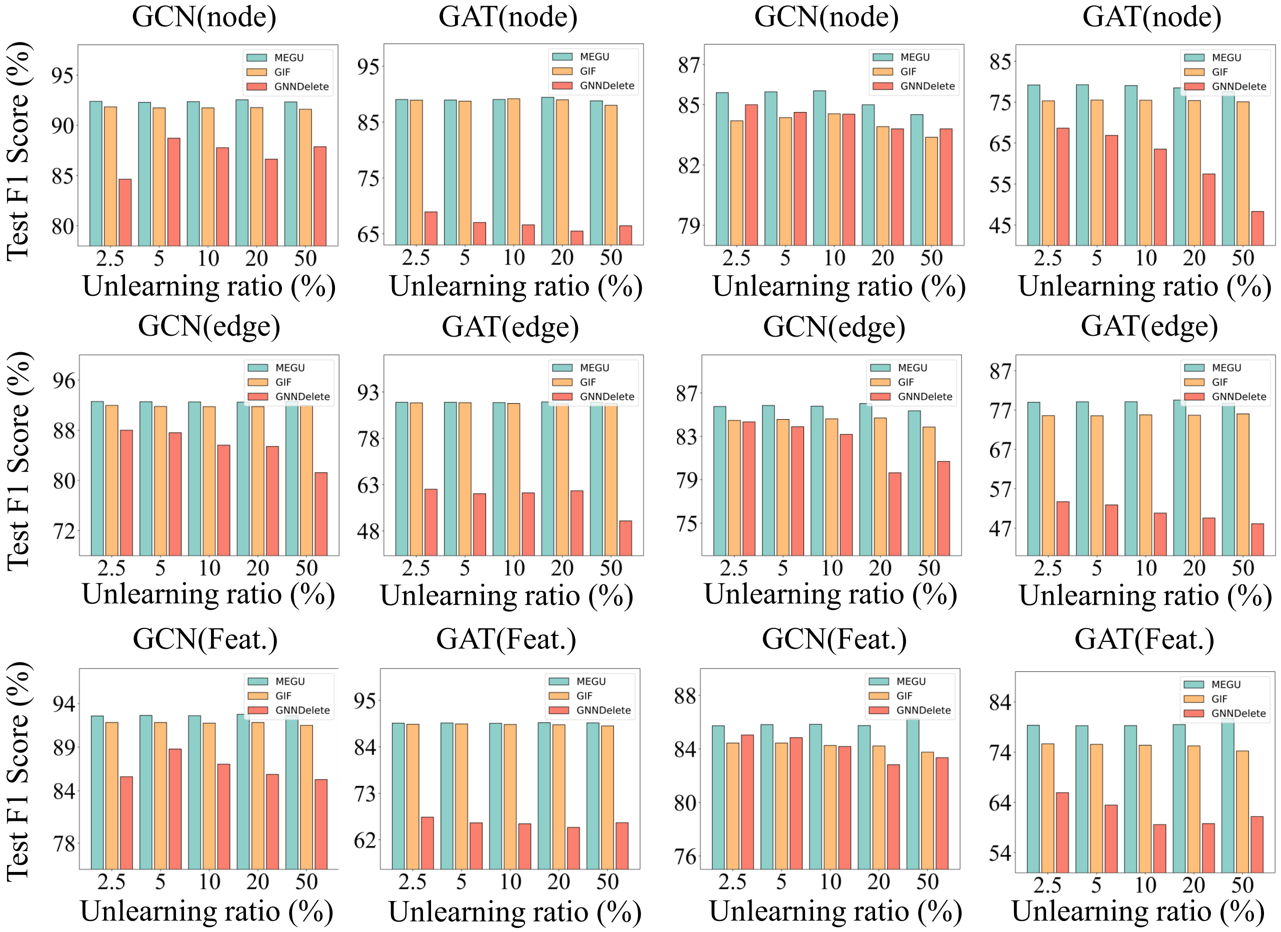
A.4 Experiment Environment

Experiments are conducted with Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz, and a single NVIDIA GeForce RTX 3090 with 24GB GPU memory. The operating system of the machine is Ubuntu 20.04.5. As for software versions, we use Python 3.8.10, Pytorch 1.13.0, and CUDA 11.7.0.

A.5 Sparsity Challenge in Feature Unlearning

In the context of feature sparsity, we posit that the feature representation of labeled nodes is partially incomplete. In the context of feature unlearning, these labeled nodes correspond to unlearning entities are afflicted by feature-related noise, while the incompleteness of feature representation aligns with the objective of feature unlearning. This necessitates mitigating the impact of unlearning features on other entities within the graph learning paradigm. In the main text, we consider masking all dimensions of features for the unlearning nodes to evaluate the feature unlearning performance of different GU strategies. However, such a choice may not encompass the entirety of the feature unlearning. To further elucidate the superior performance of MEGU in the realm of feature unlearning, we expand the experimental scope of the feature unlearning. Multiple experiments are conducted using feature masking ratios ranging from 0.1 to 1, and the obtained results are juxtaposed with those of the most competitive GIF, as illustrated in Fig. 5.

Building upon this, our findings can be summarized as follows: (1) The feature mask ratio exerts a substantial influence on GU performance. As the feature mask ratio increases, a diminishing and unstable performance trend becomes evident across various GU strategies, particularly



(a) Photo

(b) Computers

Figure 6: Performance of multiple unlearning entities and different unlearning ratios on Photo and Computer.

pronounced in the context of GIF. This phenomenon is attributed to the heightened feature mask ratio intensifying the unlearning cost within the GU framework, thereby presenting a complex trade-off between predictive accuracy for non-unlearning entities and the efficacy of forgetting unlearning entities. (2) Notable advantages of MEGU. As depicted in Fig. 5, a clear trend emerges wherein the performance of the unlearned model derived from GIF demonstrates a decline with increasing feature mask ratio. In contrast, the performance of the unlearned model obtained through MEGU maintains a consistently superior level. This resilience is attributed to MEGU's incorporation of a mutual evolution mechanism, which orchestrates a harmonious equilibrium between the predictive and unlearning modules.

A.6 Unlearning Challenges at Different Scales

In our experimental setup outlined in the main text, we adopt a default configuration wherein 10% of the graph elements are chosen as unlearning entities. In order to comprehensively evaluate the efficacy of MEGU across varying unlearning scales, we present additional experimental results in Fig. 6. According to our experiments, we observe that feature unlearning demonstrates with much less impact by the scales of unleanring tasks compared to node and edge unlearning. This discrepancy arises from the nuanced process associated with node unlearning, wherein the edges directly connected to the unlearning node are expunged, thereby disrupting the topology and inducing performance deterioration. In the edge unlearning scenarios, we remove the unlearning edges and treat the nodes connected through those edges as the entities, entailing a heightened unlearning cost. In summation, feature unlearning induces a comparatively milder impact on predictive performance for non-unlearning entities when compared to the other unlearning scenarios.

MEGU excels in accommodating diverse unlearning requests. Most evidently, as the unlearning ratio increases, an inevitable decline in the performance of the unlearned model becomes apparent across the three GU methodologies. This decline is attributed to the increased ratio of forgetten data, which thereby magnifies the negative impact on the model predictive performance, and consequently leads to a gradual erosion of the performances. In this context, it becomes evident that MEGU outperforms GNNDelete and GIF under identical unlearning conditions. These empirical findings and analyses underscore MEGU's capability to address unlearning tasks at the node, edge, and feature levels.