

Graph Unlearning Meets Influence-aware Negative Preference Optimization

Qiang Chen*
qiangchen.sh@gmail.com
Central South University
Changsha, China

Zhongze Wu*
wzz0413@csu.edu.cn
Central South University
Changsha, China

Ang He
heang@stu.shmtu.edu.cn
Shanghai Maritime University
Shanghai, China

Xi Lin†
linxi234@sjtu.edu.cn
Shanghai Jiao Tong University
Shanghai, China

Shuo Jiang
jiangshuo@tongji.edu.cn
Tongji University
Shanghai, China

Shan You
youshan@senseauto.com
SenseTime Research
Beijing, China

Chang Xu
c.xu@sydney.edu.au
University of Sydney
Sydney, Australia

Yi Chen
yichen@ust.hk
Hong Kong University of Science and
Technology
Hong Kong, China

Xiu Su†
xiusu1994@csu.edu.cn
Central South University
Changsha, China

Abstract

Recent advancements in graph unlearning models have enhanced model utility by preserving the node representation essentially invariant, while using gradient ascent on the forget set to achieve unlearning. However, this approach causes a drastic degradation in model utility during the unlearning process due to the rapid divergence speed of gradient ascent. In this paper, we introduce **INPO**, an Influence-aware Negative Preference Optimization framework that focuses on slowing the divergence speed and improving the robustness of the model utility to the unlearning process. Specifically, we first analyze that NPO has slower divergence speed and theoretically propose that unlearning high-influence edges can reduce impact of unlearning. We design an influence-aware message function to amplify the influence of unlearned edges and mitigate the tight topological coupling between the forget set and the retain set. The influence of each edge is quickly estimated by a removal-based method. Additionally, we propose a topological entropy loss from the perspective of topology to avoid excessive information loss in the local structure during unlearning. Extensive experiments conducted on five real-world datasets demonstrate that INPO-based model achieves state-of-the-art performance on all forget quality metrics while maintaining the model's utility. Codes are available at <https://github.com/sh-qiangchen/INPO>.

*Both authors contributed equally to this research.

†Both authors are corresponding authors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MM '25, Dublin, Ireland

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-2035-2/2025/10
<https://doi.org/10.1145/3746027.3754941>

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Mathematics of computing** → **Graph algorithms**.

Keywords

Graph unlearning, Negative preference optimization, Graph neural network, Fine-tuning

ACM Reference Format:

Qiang Chen, Zhongze Wu, Ang He, Xi Lin, Shuo Jiang, Shan You, Chang Xu, Yi Chen, and Xiu Su. 2025. Graph Unlearning Meets Influence-aware Negative Preference Optimization. In *Proceedings of the 33rd ACM International Conference on Multimedia (MM '25)*, October 27–31, 2025, Dublin, Ireland. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3746027.3754941>

1 Introduction

Graph-structured data[23, 26, 27, 35] play a pivotal role in multi-modal models, facilitating the discovery of relevant information among entities. To better capture the relationships, Graph Neural Networks (GNNs)[16, 20] have recently emerged as a crucial tool. With increasing awareness of privacy protection and the introduction of regulatory policies[2, 47], removing some privacy-related information from trained graph models is urgent. This has motivated a line of research on graph unlearning, aiming to strengthen *the Right to be Forgotten*. Moreover, graph unlearning is also highly valuable for removing inaccurate or outdated information contained in training samples.

Graph unlearning[5] refers to the process of forgetting or removing information related to certain features, edges and nodes from a pre-trained graph model. Designing graph unlearning models is challenging due to the strong coupling relationships between elements in graph data. Currently, most models[25, 28, 45, 51] rely on distance-based loss to preserve the predictive performance of the model on the retention set, while effectively forgetting using gradient ascent. Specially, GNNDelete[5] facilitates edge unlearning by minimizing the MSE loss between the embeddings of deleted

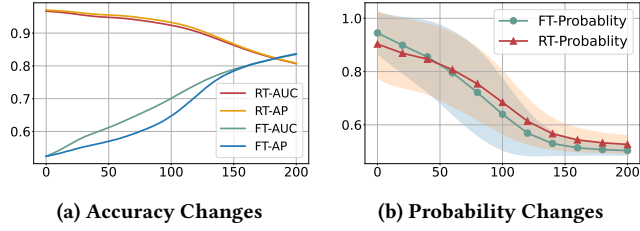


Figure 1: The accuracy and probability curve using NPO on RT and FT of DBLP. RT and FT denote the retain set and the forget set, respectively.

edges and those that were non-existent, and makes it infeasible to distinguish the representation distance between the forgot set and the retain data. Meanwhile, the model[25] based on gradient ascent exhibits **rapid divergence speed, significantly degrading model utility as unlearning progresses.**

Recently, Reinforcement Learning from Human Feedback (RLHF) offers a preference optimization manner[7, 21, 22, 60] to learn value alignment, and its superior performance has been demonstrated in crucial tasks such as LLM Unlearning and LLM Safety[6, 9, 18, 19, 44]. Direct Preference Optimization(DPO)[32] derives a straightforward approach for policy optimization by directly using preferences, thus avoiding the complexity of learning a reward function. Negative Preference Optimization(NPO)[59] ignores the positive samples used in DPO and optimizes using only negative samples, achieving a better balance between model forget quality and utility. Furthermore, NPO-based and DPO-based methods[48, 49, 53, 58] have shown excellent performance in LLM Unlearning tasks due to **slower divergence speed**, reducing the impact on model utility when executing an unlearning goal. Hence, a natural question arises: **"Are preference optimization method effective in graph unlearning tasks where data entities are strongly couple?"**

To explore this, we conduct a pilot study to investigate the impact of graph unlearning on model utility. As shown in Figure 1a, as the AUC and AP on the forget set improve, their counterparts on the retain set exhibit a corresponding decreases. Figure 1b indicates enhancing the model's ability to forget specific data instances leads to a decrement in prediction probability over previously learned data, reflecting the challenge in balancing model forget quality and utility. These two phenomena indicate that **the robustness of the model utility to the graph unlearning is insufficient.**

In this work, we propose an Influence-aware Negative Preference Optimization framework to mitigate the tight topological coupling between the forget set and the retain set, as shown in Figure 2, aiming to improve the robustness of the model utility to the unlearning process. Specifically, we first analyze the small adaptive coefficient of NPO is beneficial for the robustness and theoretically propose that unlearning high-influence edges can reduce impact on the retain set to improve the robustness, which is achieved by **enlarging the probability difference between the forget set and the retain set.** Based on this insight, we develop an influence-aware message function to amplify the influence of unlearned edges and mitigate the tight topological coupling. Our message function incorporates the influence of edges into GNN, and the influence of

each edge is quickly estimated by a removal-based method. This method is fast, requiring only a single inference, and does not impose any additional computational overhead. The proposed new message function achieves a result similar to forgetting high-impact edges. Additionally, to further preserve effective model utility, we propose a topological entropy loss function from the perspective of topology to avoid excessive information loss in the local structure before and after unlearning.

In summary, the main contributions of our paper are:

- We are the first to propose a preference optimization approach to improve the robustness of the model utility to graph unlearning.
- We theoretically propose that unlearning high-influence edges can improve the robustness and design a novel message function to amplify the effects of unlearned edges to improve the robustness.
- We propose a topological entropy loss function from the perspective of topology to avoid excessive information loss in the local structure before and after unlearning.
- We validated the effectiveness of **INPO** on five real-world datasets. The experimental results strongly indicate that INPO achieves state-of-the-art performance on all forgetting quality metrics while maintaining the model's utility. On the DBLP and Cora datasets, the performance of MI Ration improved by 6.5% and 2.3%, respectively.

2 Preliminaries

2.1 Graph Unlearning

Graph unlearning tasks consist mainly of three types: feature unlearning, node unlearning, and edge unlearning. This work focuses primarily on edge unlearning. Given a pre-trained model (i.e., the reference model) parameterized by θ_{ref} , an attributed graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ with $N = |\mathcal{V}|$ nodes, set of edges $\mathcal{E} = \{(v_i, v_j)\}_{i,j=1}^N$, and d -dimensional node features $\mathbf{X} = \{\mathbf{x}_0, \dots, \mathbf{x}_{N-1}\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ is used as dataset. Edge unlearning requires fine-tuning the pre-trained model [38, 40, 41] to forget some edges (i.e., the forget set) $\mathcal{E}_f \subseteq \mathcal{E}$ that are specified by a deletion request, while preserving performance on the retain set $\mathcal{E}_r = \mathcal{E} - \mathcal{E}_f$. In other words, we would like the unlearned model to behave as if the edges in forget set were never used to train.

2.2 Graph Neural Network

Modern GNN follow the message passing mechanism, which iteratively updates the representation of a node by aggregating representations of its neighbors. Formally, the update of node $v \in \mathcal{V}$ at GNN's i -th layer can be expressed by:

$$h_v^{(i)} = \text{ReLu}(w_0^{(i)} h_v^{(i-1)} + w_1^{(i)} \sum_{u \in N(v)} \rho_{v,u} h_u^{(i-1)}), \quad (1)$$

where $h_v^{(i)} \in \mathbb{R}^{d_i}$ is the embedding vector of node v at the i -th layer, $w_0^{(i)}$ and $w_1^{(i)}$ are weight matrices in $\mathbb{R}^{d_i \times d_{i-1}}$. The activation function for all layers is Relu [39, 42, 43], except for the last layer. $N(v)$ represents the 1-hop neighbors of node v , and ρ is the normalized weight between two nodes.

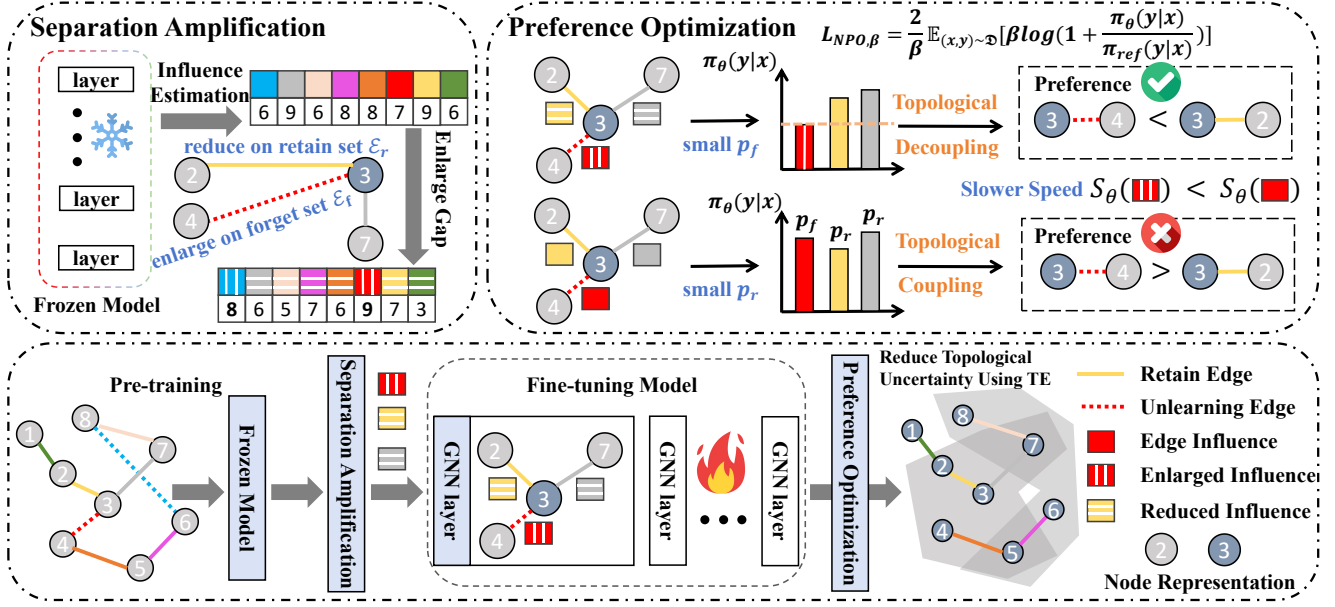


Figure 2: Overview of our INPO. Achieving graph unlearning by preference optimization considering topological decoupling.

2.3 Negative Preference Optimization

In preference optimization, only a negative response y_l is provided, the NPO loss is calculated without any positive response. Specifically, it is:

$$\mathcal{L}_{NPO,\beta}(\theta) = \frac{2}{\beta} \mathbb{E}_{(x,y_l) \sim \mathcal{D}} [\log(1 + \frac{\pi_\theta(y_l | x)}{\pi_{ref}(y_l | x)})^\beta], \quad (2)$$

Minimizing $\mathcal{L}_{NPO,\beta}$ guarantees that the prediction probability of y_l is as small as possible.

3 Robustness Against Unlearning

To the best of our knowledge, we first propose that edge unlearning can be viewed as a preference optimization problem. We treat the prediction of each edge $(\mathcal{V}_i, \mathcal{V}_j) \in \mathcal{E}_f$ as a negative response, and use NPO loss to optimize. The prediction probability of pre-trained model on the forget set is directly used as the reference policy π_{ref} . Minimizing $\mathcal{L}_{NPO,\beta}$ ensures that the prediction probability of each edge $(\mathcal{V}_i, \mathcal{V}_j) \in \mathcal{E}_f$ is as small as possible, aligning with the goal of unlearning edges in the forget set.

NPO[12, 59] indicates that the decrease of model utility is positively correlated with the model's divergence speed during unlearning, which corresponds to the gradient of the NPO loss. The gradients are as follows:

$$\nabla_\theta \mathcal{L}_{NPO,\beta} = \mathbb{E}_{\mathcal{E}_f} [S_\theta(x, y) \nabla_\theta \log \pi_\theta(y | x)], \quad (3)$$

where $S_\theta(x, y) = 2\pi_\theta^\beta(y | x) / [\pi_\theta^\beta(y | x) + \pi_{ref}^\beta(y | x)]$ can be views as an adaptive coefficient.

LEMMA 3.1. *If the predicted probability of unlearned model much less than the counterpart of original pre-trained model on the forget set, i.e., $\pi_\theta(y | x) \ll \pi_{ref}(y | x)$, the performance decrease on the retain set is slow.*

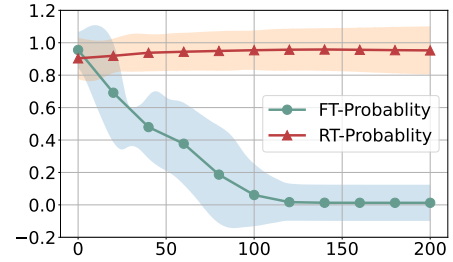


Figure 3: The predicted probability after unlearning high-influence edges on DBLP.

According to equation 3, we know that $S_\theta(x, y) \ll 1$ when $\pi_\theta(y | x) \ll \pi_{ref}(y | x)$, which means NPO diverge much slower than GA loss ($\nabla_\theta \mathcal{L}_{GA} = \mathbb{E}_{\mathcal{E}_f} [\nabla_\theta \log \pi_\theta(y | x)]$). However, as the probability of forgetting edges decreases, **the probability of retaining edges also decreases due to topological coupling in graph unlearning task.**

A considerable corpus of research[4, 13, 29, 46, 50, 61] has substantiated that nodes or edges with high influence lead to better performance in link prediction tasks, i.e., the predicted probability for the existence of unlearned edge is relatively low compared to an counterpart on the retain set as unlearning progresses. Therefore, amplifying the influence of unlearned edges to **enlarge the probability difference**(we give theoretical proof in Supplementary Materials A.4) between the forget set and the retain set can mitigate topological coupling, i.e., $\pi_\theta^\beta(\mathcal{E}_f) < \pi_\theta^\beta(\mathcal{E}_d)$, to achieve robustness against unlearning on graph.

PROPOSITION 3.2. *For edge unlearning, edges with high influence exhibit a low predicted probability $\pi_\theta(y | x)$, and unlearning these*

edges would lead to slower divergence speed and reduced impact on the retain set.

Proposition 3.2 indicates **NPO is suitable for handling requests that contain a significant number of high-influence edges on edge unlearning task**. To validate the proposition, this work configures a deletion request to consist of edges characterized by high influence. As shown in Figure 3, as the predicted probability of high-influence edges on the forget set decreases, the counterpart on the retain set remains stable without decreasing.

4 Methodology

Although NPO is suitable for high-influence edge unlearning, random deletion requests are more common. Improving the general unlearning capability of NPO is challenging for graph unlearning. In this section, we propose INPO to improve the robustness of the model utility to graph unlearning.

4.1 Fast Estimation of Edge Influence

In this work, we use a remove-based approach[33, 37, 56, 57] to estimate the influence of the nodes. Subsequently, this assessment enables us to determine the influence exerted by the edges within the graph. To express the influence of the node $v_r \in \mathcal{V}$, we define it as:

$$F_{g_\theta}(v_r) = \sum_{i=1, i \neq r}^N \|g_\theta(G)_i - g_\theta(G_{-v_r})_i\|_1, \quad (4)$$

where $g_\theta(G)_i \in \mathbb{R}^c$ (c is the number of classes) denote the predicted class probability of node v_i , and is trained on graph G . G_{-v_r} is the graph that node v_r is removed.

To obtain the influence of all nodes, a direct and simple method is to alternately remove every node and predict with the trained GNN on the modified graph. The difference is the influence of all nodes. However, this brute-force way is time-consuming. Considering efficiency, we adopt the gradient information to approximate the removal-based node influence as NORA[24]. NORA derives the node influence as follows:

$$\begin{aligned} F_{g_\theta}(v_r) &= \sum_{i=0}^{L-1} (\hat{d}_r^{(L-1-i)} \hat{h}_r^{(i)}) + k_3 \cdot \delta Topo_r, \\ \hat{d}_r &= 1 - \frac{d_r}{(N-1)(d+\gamma)}, \quad \hat{h}_r^{(i)} = \frac{d_r}{d_r + \gamma} \left\| \left(f_r \frac{\partial f_r}{\partial h_r^{(i)}} \right) \circ h_r^{(i)} \right\|_1, \\ \delta Topo_r &= \sum_{i \in N(r)} \sum_{j \in N(i)} [k_1 \left(\frac{1}{\sqrt{d_i - 1}} - \frac{1}{\sqrt{d_i}} \right) \\ &\quad + (1-k_1) \left(\frac{1}{d_i - 1} - \frac{1}{d_i} \right)] [k_2 \frac{1}{\sqrt{d_j}} + k'_2 \frac{1}{d_j} + (1-k_2 - k'_2)], \end{aligned} \quad (5)$$

where d_r and d represent the degree of the removed node v_r and the average degree of the entire graph, respectively. $f_r = \sum_{i=1, i \neq r}^N h_i^{(L)}$ is the sum of the predicted probability of all nodes except node v_r at L -th GNN layer, and k_1, k_2, k'_2, k_3 and γ are hyperparameters. $\circ$ denotes element-wise production.

According to equation 5, we can get the influence of all nodes for the entire graph only by a single inference. The edge (v_i, v_j) influence can be expressed as:

$$\xi_{ij} = F_{g_\theta}(v_i) + F_{g_\theta}(v_j). \quad (6)$$

Compared to the brute-force method, this gradient approximation can make a fast estimation without excessive training and computational overhead.

4.2 Influence-Enhanced MPNN

To improving the general unlearning capability of NPO, i.e., the deletion request consists of a randomly selected subset of edges, we redesign the message passing mechanism in GNN to adapt to edge unlearning task. The traditional message passing mechanism contains three components[14]: the message function, the aggregate function, and the update function. The proposition 3.2 shows that NPO is suitable for unlearning edges with high influence. A direct design is overwriting the message function and enhancing the influence of low-influence edges on the forget set.

The common message function is:

$$m_{vu}^{(l)} = \rho_{vu} h_u^{(l-1)}, \quad (7)$$

where $m_{vu}^{(l)}$ is the message at GNN layer l , and $\rho_{vu} = \frac{1}{\sqrt{d_v} \sqrt{d_u}}$ is normalized weight between two nodes.

Before fine-tuning the pre-trained GNN for edge unlearning, we first use the NORA algorithm in Section 4.1 to estimate the influence of all edges. Therefore, overwriting the message function of unlearned model would not affect the estimation of edge influence in the graph.

After obtaining the influence of each edge through NORA, we rewrite the message function as follows:

$$m_{vu}^{(l)} = e^{q\xi_{vu}} \rho_{vu} h_u^{(l-1)}, \quad (v, u) \in \mathcal{E}_f, \quad (8)$$

where q is a hyperparameter. Compared to influence of edges on the retain set, the new message function enhances the influence on the forget set. In the actual implementation, the size of unlearned edges is small, and we adopt another method to reduce the influence of edges in the retain set. The the message function is:

$$m_{vu}^{(l)} = e^{-q\xi_{vu}} \rho_{vu} h_u^{(l-1)}, \quad (v, u) \in \mathcal{E}_r. \quad (9)$$

The new message considers the influence of edges in the original graph and reduces impact on the retain set by amplifying the influence of unlearned edges, which makes the NPO suitable for edge unlearning. Actually, this approach mitigates the tight topological coupling by enlarging the probability difference between the forget set and the retain set.

4.3 Topological Entropy

This work focuses on parameter optimization, i.e., **unlearning fine-tuning**[11], and modifies the pre-trained model parameters by different objectives. Based on the objective of unlearned graph model, we categorize existing methods[31, 36] into two paradigms: **preserve model utility** and **improve forget quality**.

For preserving utility of model, We consider the following two methods:

- **Gradient Descent (GD)** simply uses the training CE loss to perform gradient descent on the retain set, as follows:

$$L_{GD}(\mathcal{E}_r; \theta) = \mathbb{E}_{(x,y) \sim \mathcal{E}_r} [-\log \pi_\theta(y | x)]. \quad (10)$$

Table 1: Complexity comparison

Method	Fine-tuning	Ours
Time	$LEF + LNF^2$	$2LEF + LNF^2$
Space	$E + LF^2 + LNF$	$2E + LF^2 + LNF$

- **Kullback-Leibler Divergence (KL)**[15] is to minimize the difference of the prediction distribution of the unlearned model and the reference model on the retain set, as follows:

$$L_{KL}(\mathcal{E}_r; \theta) = \mathbb{E}_{(x,y) \sim \mathcal{E}_r} [KL(\pi_\theta(y | x) || \pi_{ref}(y | x))]. \quad (11)$$

For improving forget quality, We consider the following two methods:

- **Gradient Ascent (GA)** maximize the CE loss loss on the forget set, as follows:

$$L_{GA}(\mathcal{E}_f; \theta) = -\mathbb{E}_{(x,y) \sim \mathcal{E}_f} [-\log \pi_\theta(y | x)]. \quad (12)$$

- **Direct Preference Optimization (DPO)** use prediction on the forget set as negative samples and random prediction on the retain set as positive samples.

To incorporate the properties of the graph into the edge unlearning process, we propose a new optimization objective from a topological perspective. Inspired by the Neighborhood Influence property which the embedding of the neighboring subgraph remains largely unchanged before and after the edge deletion in GNNDelete. We directly average embedding h_i and h_j at layer L to obtain the distribution of the local subgraph around that edge e_{ij} , it is:

$$G_{ij} = \frac{1}{2}(h_i^{(L)} + h_j^{(L)}), \quad (13)$$

where G_{ij} represents the embedding of the L -hop local structure.

To ensure that edge unlearning does not cause significant changes to the neighboring nodes, we propose topological entropy as an optimization objective as following:

$$TE_{ij} = -\sum G_{ij}^{ref} \log(G_{ij}), \quad e_{ij} \in \mathcal{E}_f, \quad (14)$$

where G_{ij}^{ref} represents the pre-trained embedding of the L -hop local structure.

Finally, We employ a holistic loss function to optimize two losses:

$$Loss = \lambda_1 L_{NPO} + \lambda_2 GD + \lambda_3 TE, \quad (15)$$

where $\lambda_1, \lambda_2, \lambda_3$ are weights associated with forget quality and model utility. These weights decide whether the model is more inclined to improve forget quality or preserve model utility.

4.4 Complexity Analysis

Compared to other unlearning fine-tuning mdoels, our model adds a single inference to calculate the influence of edges, and this overhead is negligible. We list the time and space complexities[10, 52] in Table 1. E denotes the number of edges, and F denotes the feature dimension. It is easy to see that the order of complexity remains unchanged, the time complexity and space complexity are $O(LEF + LNF^2)$ and $O(E + LF^2 + LNF)$, respectively.

5 Experiments

5.1 Experimental Settings

5.1.1 Datasets. To thoroughly validate the effectiveness of our model and ensure a comprehensive generalization evaluation, we used five real-world datasets[1, 17]: Cora, PubMed, DBLP, CS, OGB-Collab.

5.1.2 Baseline Models. In our experiments, we select 4 advanced and 5 self-designed methods based on the loss combination discussed in Section 4.3 as baselines for performance comparison. The description of these baselines is as follows.

Advanced Fine-tuning Methods for Edge Unlearning.

- **Retrain**[28]. This method, while straightforward, is inefficient as it requires retraining models from scratch to unlearn specific edges.
- **GIF**[51]. This method accurately estimates parameter changes by designing influence functions to directly modify the parameters for edge unlearning.
- **GNNDelete**[5]. This method achieves unlearning by approximating the representation of edges to be forgotten to those that did not exist in the pretrained model, while keeping the neighbors' representations minimally changed.
- **UtU**[45]. Compared to GNNDelete, it only uses the graph after edge deletion for a single inference.

Self-designed Fine-tuning Methods for Edge Unlearning.

- **GA+GD**[31]. This method use GA loss on the forget set and GD loss on the retain set as optimization objective.
- **GA+KL**[31]. This method use GA loss on the forget set and KL loss on the retain set as optimization objective.
- **DPO**[32]. This method treat the edge unlearning as a preference optimization problem. We use predicted probability on the forget set as negative samples and random probability on the retain set as positive samples to perform preference optimization.
- **DPO+GD**[58]. This method use DPO loss and GD loss on the retain set as optimization objective.
- **DPO+KL**[58]. This method use DPO loss and KL loss on the retain set as optimization objective.

5.1.3 Evaluation Metrics. To measure the effectiveness of our model, we use model utility and forget quality as metrics. The model utility refers to its ability to maintain the original inference capability after unlearning, measured by AUC and AP on the retain set. The forget quality measured by AUC and AP on the forget set. $AP = \sum_n (R_n - R_{n-1}) \cdot P_n$ where P_n and R_n are the precision and recall at the n -th threshold. AUC is the area under the Receiver Operating Characteristic Curve. To evaluate whether the model has truly achieved forgetting, we use **the probability of edge($e \in \mathcal{E}_f$) existence** after unlearning as another metric of forgetting quality, i.e., $p_f(avg)$. MI Ratio[30] is a commonly used metric for measuring model forget quality, which quantifies the success rate of a Membership Inference (MI) attack[34, 55], by calculating the ratio of presence probability of $\mathcal{E}_f$ before and after the deletion operator.

5.1.4 Implementation Details. We evaluate the effectiveness of our model on edge unlearning tasks. We perform experiments on two settings: (1) the deletion request consists of edges with high

Table 2: Comparison results of our model with self-designed fine-tuning methods. In each column, the best result is indicated in red, while the runner-up result is marked with blue. The pre-training based on link prediction task.

Model	DBLP					Cora				
	$\mathcal{E}_r$		$\mathcal{E}_f$			$\mathcal{E}_r$		$\mathcal{E}_f$		
	AUC	AP	AUC	AP	MI Ratio	AUC	AP	AUC	AP	MI Ratio
GA	0.6787	0.6156	0.6046	0.5589	1.30	0.5106	0.5799	0.6025	0.5766	2.77
GA+GD	0.6122	0.6038	0.8498	0.8445	2.13	0.5181	0.5851	0.6153	0.5923	2.74
GA+KL	0.6788	0.6157	0.6046	0.5589	1.30	0.6668	0.6125	0.5487	0.5266	1.21
DPO	0.6501	0.6718	0.8639	0.8312	2.60	0.5495	0.6008	0.6718	0.6543	2.76
DPO+GD	0.9432	0.9352	0.4842	0.4864	1.02	0.7256	0.7141	0.6311	0.5706	1.54
DPO+KL	0.8245	0.7713	0.4657	0.4819	1.00	0.7104	0.6434	0.5046	0.5023	1.00
NPO	0.9002	0.9027	0.7913	0.8038	1.79	0.8996	0.9015	0.7142	0.7036	1.84
INPO	0.8853	0.8852	0.9037	0.9010	1.59	0.8973	0.8916	0.9058	0.8885	1.61

Table 3: Comparison results of our model with advanced fine-tuning methods. In each column, the best result is indicated in red, while the runner-up result is marked with blue, and the third place is marked with orange. Evaluation: link prediction.

Model	DBLP					Cora				
	$\mathcal{E}_r$		$\mathcal{E}_f$			$\mathcal{E}_r$		$\mathcal{E}_f$		
	AUC	AP	AUC	AP	MI Ratio	AUC	AP	AUC	AP	MI Ratio
Retrain	0.9614	0.9645	0.5153	0.5131	1.05	0.9364	0.9355	0.4818	0.4867	1.09
GA	0.6787	0.6156	0.6046	0.5589	1.30	0.5106	0.5799	0.6025	0.5766	2.77
GIF	0.9688	0.9714	0.5217	0.5168	1.03	0.9678	0.9668	0.4913	0.4937	1.03
GNNDelete	0.9573	0.9601	0.9731	0.9754	1.69	0.9609	0.9609	0.9797	0.9834	1.75
UtU	0.9687	0.9714	0.5158	0.5098	1.03	0.9677	0.9668	0.4965	0.4924	1.03
NPO	0.9002	0.9027	0.7913	0.8038	1.79	0.8996	0.9015	0.7142	0.7036	1.84
INPO	0.8853	0.8852	0.9037	0.9010	1.59	0.8973	0.8916	0.9058	0.8885	1.61
INPO-S	0.9533	0.9554	0.9809	0.9823	1.80	0.9613	0.9613	0.9802	0.9836	1.79

influence and (2) the deletion request consists of random edges. To perform edge unlearning tasks, The proportion of edges we delete is 0.5%. We conducted all experiments 5 times and reported average value, ignored the variance because they were extremely small.

5.2 Overall Performance Experiments

Analysis on the baselines. In Table 2, 3 and 4, we summarize the overall performance of INPO and the baselines. We observe that the forget quality of DPO and NPO constantly surpasses most advanced fine-tuning methods, showing the effect of preference optimization for graph edge unlearning. Also, we found that GD loss can improve the performance on the retain set. Moreover, most baselines is hard to strike a balance between model utility and the quality of forgetting, except for GNNDelete. An interesting observation is that the MI ratio and $\frac{pr}{pf}$ of GNNDelete are relatively low, indicating that it does not truly unlearn the edges on the forget set. The higher edge prediction probability on the forget set also indicates this. In conclusion, the limitations of baselines hinder their ability to achieve consistent success.

The effectiveness of INPO. Overall, INPO outperforms most of the baselines in terms of model utility and forget quality, and INPO-S achieves state-of-the-art performance on all forget quality

Table 4: The average predicted probability of our model with baseline methods on the retain set and the forget set. Table 4 has all hyper-parameters consistent with those in Table 3. In each column, the best result is indicated in red, while the runner-up result is marked with blue.

Model	DBLP			Cora		
	pf	pr	$\frac{pr}{pf}$	pf	pr	$\frac{pr}{pf}$
Retrain	0.9305	0.9010	0.97	0.8971	0.8626	0.96
GIF	0.9504	0.9129	0.96	0.9492	0.9093	0.96
GNNDelete	0.5814	0.8517	1.46	0.5595	0.8524	1.52
UtU	0.9496	0.9122	0.96	0.9472	0.9076	0.96
NPO	0.5475	0.5910	1.08	0.5323	0.5476	1.03
INPO	0.6154	0.8143	1.32	0.6070	0.8036	1.32
INPO-S	0.5451	0.8388	1.54	0.5473	0.8562	1.56

metrics while maintaining the model’s utility. In particular, we obtain large forget quality gains over the best baseline in two datasets by 6.5% and 2.3% for MI Ratio, respectively. Additionally, compared to GNNDelete, which was previously the baseline with the best balance between model utility and the quality of forgetting, INPO-S improves $\frac{pr}{pf}$ by 5.5% and 2.6% on two datasets. Notably, INPO-S

Table 5: Ablation results of our model.

Model	DBLP					Cora				
	$\mathcal{E}_r$		$\mathcal{E}_f$			$\mathcal{E}_r$		$\mathcal{E}_f$		
	AUC	AP	AUC	AP	MI Ratio	AUC	AP	AUC	AP	MI Ratio
NPO+GD	0.8106	0.8234	0.8549	0.8167	1.91	0.7923	0.7809	0.9057	0.8934	1.73
NPO+IMPNN	0.8708	0.8693	0.8150	0.8248	1.92	0.8375	0.8449	0.7822	0.7845	1.93
NPO+TE	0.8977	0.8966	0.8323	0.8501	1.86	0.8734	0.8767	0.7540	0.7561	1.90
NPO+GD+IMPNN	0.8231	0.8253	0.8792	0.8789	1.86	0.8543	0.8456	0.8847	0.8716	1.61
NPO+GD+TE	0.8384	0.8362	0.9220	0.9227	1.76	0.8562	0.8465	0.9103	0.8954	1.72
INPO	0.8853	0.8852	0.9037	0.9010	1.59	0.8973	0.8916	0.9058	0.8885	1.61

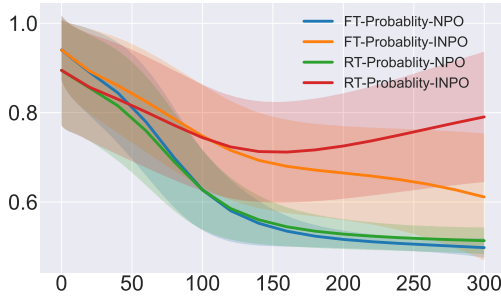


Figure 4: The prediction probability change curve of NPO and INPO on Cora validation dataset.

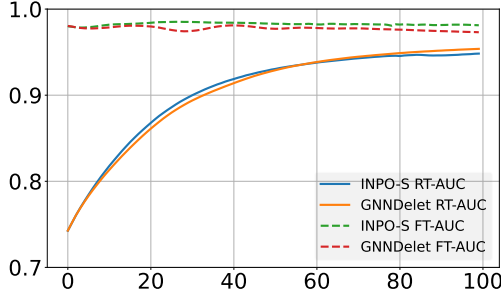


Figure 5: The AUC change curve of INPO-S and GNNDelate on retain set and forget set for DBLP validation dataset.

achieves a perfect model utility that is essentially the same as GNNDelate. This evidence suggests that INPO is able to achieve SOTA edge unlearning, and the decreased prediction probability for edges on the forget set provide a specific explanation. More results of experiments are given in Supplementary Materials A.2.

Comparison between NPO and INPO. As shown in Figure 4, we found that INPO effectively mitigates the impact of the unlearning process on model utility, which is missing in NPO. From the gradient perspective, IMPNN reduces the adaptive coefficient $S_\theta(x, y)$, thereby minimizing impact on model utility, while maintaining model utility through TE loss. **The substantial improved $\frac{pr}{pf}$ indicates effectiveness of mitigating the tight topological coupling.** Further experimental results are provided in the ablation study and Supplementary Materials A.2(Figure 3).

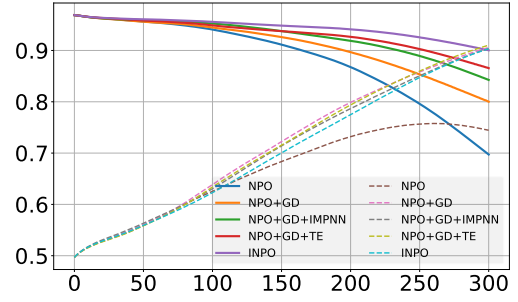


Figure 6: The AUC change curve of all ablation models on Cora validation dataset. The solid lines denotes AUC on retain set and the dashed lines represents AUC on forget set.

Comparison between GNNDelate and INPO-S. INPO-S refers to a method that incorporates additional parameters **initialized to zero** for forgetting, similar to GNNDelate. The key distinction is that INPO-S does not utilize the **Deleted Edge Consistency** loss employed by GNNDelate for the forgetting process. As shown in Figure 5, we found that INPO performs better and more stably in maintaining the forgetting capability.

5.3 Ablation Experiment

Here we empirically dissect the contribution of (1) GD loss, (2) redesigned MPNN, and (3) topological entropy regularization. We proposed five ablations models respectively:

- **NPO-GD**, which uses GD loss as a regularization term for NPO loss.
- **NPO-IMPNN**, which replaces the message passing mechanism with influence-based message function.
- **NPO-TE**, which uses topological entropy as a regularization term for NPO loss.
- **NPO-GD-IMPNN**, which uses GD loss regularization and influence-based message function.
- **NPO-GD-TE**, which uses GD loss and topological entropy regularization.

Ablation results. In Table 5, we report the ablation results. By comparing NPO+GD and NPO+GD+IMPNN, we discover that can effectively maintain AUC and AP on the retain set while enhancing the forget quality on the forget set for fine-tune. This result

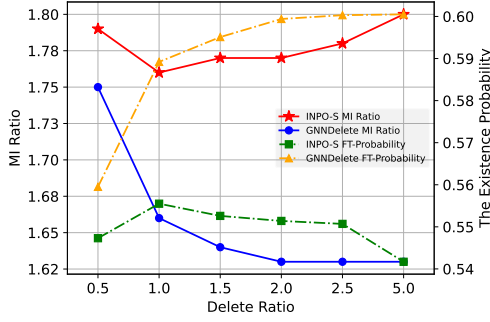


Figure 7: MI Ratio performance and the probability of edge ($e \in \mathcal{E}_f$) existence for different delete ratio(%), and a lower probability of edge existence indicates better unlearning.

demonstrates the effectiveness of redesigned influence-based message function. Further, the comparison between NPO+GD and NPO+GD+TE implies that, as the unlearn process progresses, it can still effectively maintain performance on the retain set. This result demonstrates the effectiveness of topological entropy regularization. Overall, these three ablation models justify the efficacy of our framework.

Trade-off between utility and forget quality. Here we are interested in the changes of utility and forget quality during the optimization process, and we visualize the trade-off process on Cora dataset. As shown in Figure 6, we observe that: (1) In terms of model utility and the forget quality, the three ablation models and INPO significantly outperform the original NPO. (2) INPO is largely consistent with the three ablation models (except for NPO) for the forget quality, but it excels in maintaining AUC without large decline.

5.4 Robustness Analysis Experiments

In this section, we delve into the robustness of our framework from three perspectives: β in Equation 2, λ_1 in Equation 15 and λ_3 in Equation 15. The analysis was conducted using the Cora dataset.

The robustness to different delete ratio. As shown in Figure 7, INPO-S significantly outperforms the current best baseline in both MI Ratio and the probability of edge existence. It is noteworthy that our MI Ratio not only outperforms the baseline, but also **does not show a decline as the delete ratio increases**, unlike GNNDelete. In fact, our model demonstrates a slight improvement with higher delete ratio. Additionally, **the probability of the edges we aim to forget does not increase as the delete ratio grows**, indicating that INPO-S achieves true forgetting even at higher delete ratio. These findings indicate that our model exhibits strong robustness across different deletion ratios.

The impact of β . In Figure 8, We show the impact of the hyper-parameter β on INPO's performance, i.e., AUC on the retain set and forget set. We observe that as the number of β increases, the AUC on the retain set gradually improves. However, the AUC on the forget set reaches a plateau when $\beta = 5$, and then begins to decline. This difference is caused by the divergence speed, which is related to the adaptive coefficient $S_\theta(x, y) = 2\pi_\theta^\beta(y | x) / [\pi_\theta^\beta(y | x) + \pi_{ref}^\beta(y | x)]$.

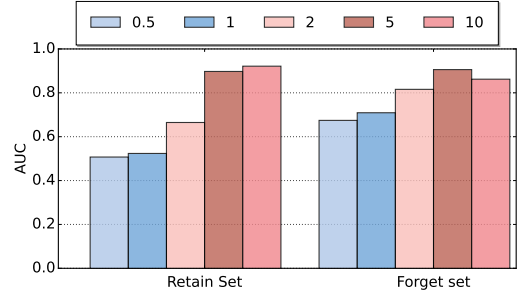


Figure 8: AUC Performance on the retain set and the forget set at different β .

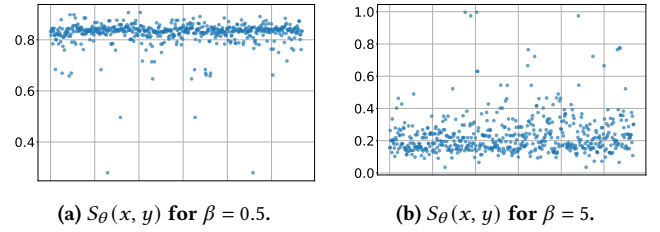


Figure 9: The coefficient $S_\theta(x, y)$ at epoch 200 for $\beta = 0.5$ and $\beta = 5$. A data point represents an edge to be forgotten.

When β is small, the divergence rate of the entire process becomes too rapid, as π_{ref} is directly obtained from the pre-trained model. As shown in Figure 9, the divergence speed is fast when $\beta = 0.5$, which would lead to the model utility decreasing quickly. On the other hand, an overly large β leads to an excessively low divergence speed, which also results in a decline in the forget quality.

The impact of λ_1 . As shown in Supplementary Materials Figure 1, both too small and too large λ_1 can lead to poor AUC on forget set. An overly small NPO loss can lead to ineffectiveness of our model, thereby preventing the model from unlearning.

The impact of λ_3 . As depicted in Supplementary Materials Figure 2, owing to the incorporation of **topological entropy regularization** in INPO, we investigated the influence of TE loss on model performance. The results affirm the robustness of TE loss in our model, and values between 0.5 and 0.8 are all reasonable. Different values of λ_3 have little difference on overall performance, with only larger values of λ_3 causing a slight decrease in AUC on the forget set. In summary, topological entropy regularization is useful and robust.

6 Conclusion

To improve the robustness of the model utility to the unlearning process, we propose INPO that amplify the effects of low-influence edges on the forget set to achieve topological decoupling and topological entropy loss to avoid excessive information loss in the local structure during unlearning. Extensive experiments conducted on five real-world datasets demonstrate effectiveness of our model and achieve SOTA performance on all forget quality metrics.

Acknowledgments

This research is funded by National Natural Science Foundation of China (No. 62406347 and No. 62202302). This research is supported by the Hong Kong RGC Theme-based Research Scheme T32-615-24/R.

References

- [1] Aleksandar Bojchevski and Stephan Günnemann. 2018. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *ICLR*.
- [2] CCPA. 2018. California Consumer Privacy Act of 2018. https://leginfo.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375. AB-375, Signed into law on June 28, 2018.
- [3] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. 2022. Graph unlearning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*.
- [4] Xiaolong Chen, Yifan Song, and Jing Tang. 2024. Link Recommendation to Augment Influence Diffusion with Provable Guarantees. In *Proceedings of the ACM Web Conference 2024 (WWW '24)*. Association for Computing Machinery, 2509–2518.
- [5] Jiali Cheng, George Dasoulas, Huan He, Chirag Agarwal, and Marinka Zitnik. 2023. GNNDelete: A general unlearning strategy for graph neural networks. In *ICLR*.
- [6] Bai Chenjia, Zhang Yang, Qiu Shuang, Zhang Qiaosheng, Xu Kang, and Li Xue-long. 2025. ONLINE PREFERENCE ALIGNMENT FOR LANGUAGE MODELS VIA COUNT-BASED EXPLORATION. In *ICLR*.
- [7] Li Chenliang, Zeng Siliang, Liao Zeyi, Li Jiaxiang, Kang Dongyeop, Garcia Alfredo, and Hong Mingyi. 2025. LEARNING REWARD AND POLICY JOINTLY FROM DEMONSTRATION AND PREFERENCE IMPROVES ALIGNMENT. In *ICLR*.
- [8] Weilin Cong and Mehrdad Mahdavi. 2023. GraphEditor: An efficient graph representation learning and unlearning approach.
- [9] Juntao Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2024. SAFE RLHF: SAFE REINFORCEMENT LEARNING FROM HUMAN FEEDBACK. In *ICLR*.
- [10] Blakely Derrick, Lanchantin Jack, and Qi Yanjun. 2019. Time and Space Complexity of Graph Convolutional Networks. (2019). <https://api.semanticscholar.org/CorpusID:269411067>
- [11] Hu Edward J, shen yelong, Wallis Phillip, Allen-Zhu Zeyuan, Li Yuanzhi, Wang Shean, Wang Lu, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*.
- [12] Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. 2024. Simplicity Prevails: Rethinking Negative Preference Optimization for LLM Unlearning. In *NeurIPS*.
- [13] Lei Fan, Dongdong Fan, Zhiguang Hu, Yiwen Ding, Donglin Di, Kai Yi, Maurice Pagnucco, and Yang Song. 2025. Manta: A large-scale multi-view and visual-text anomaly detection dataset for tiny objects. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 25518–25527.
- [14] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*. PMLR, 1263–1272.
- [15] Rajdeep Halder, Wang Ziyi, Song Qifan, Lin Guang, and Xing Yue. 2025. LLM Safety Alignment is Divergence Estimation in Disguise. *arXiv preprint arXiv:2502.00657* (2025).
- [16] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.
- [17] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: datasets for machine learning on graphs. In *NeurIPS*.
- [18] Jiaming Ji, Donghai Hong, Borong Zhang, Boyuan Chen, Josef Dai, Boren Zheng, Tianyi Qiu, Boxun Li, and Yaodong Yang. 2024. PKU-SafeRLHF: Towards Multi-Level Safety Alignment for LLMs with Human Preference. In *NeurIPS*.
- [19] Wu Junkang, Xie Yuexiang, Yang Zhengyi, Wu Jiancan, Chen Jiawei, Gao Jinyang, Ding Bolin, Wang Xiang, and He Xiangnan. 2025. TOWARDS ROBUST ALIGNMENT OF LANGUAGE MODELS: DISTRIBUTIONALLY ROBUSTIFYING DIRECT PREFERENCE OPTIMIZATION. In *ICLR*.
- [20] Thomas N. Kipf and Welling Max. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.
- [21] Aaron Jiaxun Li, Satyapriya Krishna, and Himabindu Lakkaraju. 2025. Joint Reward and Policy Learning with Demonstrations and Human Feedback Improves Alignment. In *ICLR*.
- [22] Chenliang Li, Siliang Zeng, Zeyi Liao, Jiaxiang Li, Dongyeop Kang, Alfredo Garcia, and Hong Mingyi. 2025. Joint Reward and Policy Learning with Demonstrations and Human Feedback Improves Alignment. In *ICLR*.
- [23] Jianing Li, Chaoqun Yang, Guanhua Ye, and Quoc Viet Hung Nguyen. 2024. Graph neural networks with deep mutual learning for designing multi-modal recommendation systems. *Information Sciences* (2024).
- [24] Weikai Li, Zhiping Xiao, Xiao Luo, and Yizhou Sun. 2024. Fast Inference of Removal-Based Node Influence. In *Proceedings of the ACM Web Conference 2024 (WWW '24)*. Association for Computing Machinery, New York, NY, USA.
- [25] Xunkai Li, Yulin Zhao, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, and Guoren Wang. 2024. Towards Effective and General Graph Unlearning via Mutual Evolution. In *AAAI*, Vol. 38. 13682–13690.
- [26] Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, and Xinwang Liu. 2024. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multi-modal. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024).
- [27] Wanying Liang, Pasquale De Meo, Yong Tang, and Jia Zhu. 2024. A survey of multi-modal knowledge graphs: Technologies and trends. *Comput. Surveys* 56, 11 (2024), 1–41.
- [28] Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. 2022. The Right to be Forgotten in Federated Learning: An Efficient Realization with Rapid Retraining. In *IEEE Conference on Computer Communications*. 2022b.
- [29] Zemin Liu, Trung-Kien Nguyen, and Yuan Fang. 2021. Tail-GNN: Tail-Node Graph Neural Networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 1109–1119.
- [30] Iyiola E. Olutunji, Wolfgang Nejdl, and Megha Khosla. 2021. Membership inference attack on graph neural networks. In *Proceedings of the IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications*.
- [31] Maini Pratyush, Feng Zhili, Schwarzschild Avi, Lipton Zachary C., and Zico Kolter J. 2024. TOFU: A Task of Fictitious Unlearning for LLMs. In *NeurIPS*.
- [32] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *NeurIPS*.
- [33] Yao Rong, Guanchu Wang, Qizhang Feng, Ninghao Liu, Zirui Liu, Enkelejda Kasneci, and Xia Hu. 2023. Efficient GNN Explanation via Learning Removal-based Attribution. *arXiv:2306.05760* [cs.LG]
- [34] Alexandre Sablayrolles, Matthijs Douze, Yann Olivier, Cordelia Schmid, and Hervé Jégou. 2019. White-box vs black-box: Bayes optimal strategies for membership inference. In *Proceedings of the International Conference on Machine Learning*.
- [35] Lei Shi, Jiapeng Yang, Pengtao Lv, Lu Yuan, Feifei Kou, Jia Luo, and Mingying Xu. 2024. Self-derived Knowledge Graph Contrastive Learning for Recommendation. In *Proceedings of the 31st ACM International Conference on Multimedia (MM'23)*. Association for Computing Machinery, New York, NY, USA, 7571–7580.
- [36] Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogang Liu, Luke Zettlemoyer, Noah A. Smith, and Chiyuan Zhang. 2025. MUSE: Machine Unlearning Six-Way Evaluation for Language Models. In *ICLR*.
- [37] Ilia Stepin, Jose M. Alonso, Alejandro Catala, and Martín Pereira-Fariña. 2021. A Survey of Contrastive and Counterfactual Explanation Generation Methods for Explainable Artificial Intelligence. *IEEE Access* 9 (2021), 11974–12001. doi:10.1109/ACCESS.2021.3051315
- [38] Xiu Su, Tao Huang, Yanxi Li, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. 2021. Prioritized architecture sampling with monte-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10968–10977.
- [39] Xiu Su, Shan You, Tao Huang, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. 2021. Locally free weight sharing for network width search. *arXiv preprint arXiv:2102.05258* (2021).
- [40] Xiu Su, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. 2021. Bcnet: Searching for network width with bilaterally coupled network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2175–2184.
- [41] Xiu Su, Shan You, Jiyang Xie, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. 2022. Searching for network width with bilaterally coupled network. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 7 (2022), 8936–8953.
- [42] Xiu Su, Shan You, Jiyang Xie, Minghai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. 2022. ViTAS: Vision transformer architecture search. In *European Conference on Computer Vision*. Springer, 139–157.
- [43] Xiu Su, Shan You, Minghai Zheng, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. 2021. K-shot nas: Learnable weight-sharing for nas with k-shot supernet. In *International Conference on Machine Learning*. PMLR, 9880–9890.
- [44] Cha Sungmin, Cho Sungjun, Hwang Dasol, and Lee Moontae. 2025. TOWARDS ROBUST AND PARAMETER-EFFICIENT KNOWLEDGE UNLEARNING FOR LLMs. In *ICLR*.
- [45] Jiajun Tan, Fei Sun, Ruichen Qiu, Du Su, and Huawei Shen. 2024. Unlink to Unlearn: Simplifying Edge Unlearning in GNNs. In *Companion Proceedings of the ACM Web Conference 2024 (WWW '24)*. Association for Computing Machinery, New York, NY, USA, 489–492.
- [46] Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Yiqi Wang, Jiliang Tang, Charu Aggarwal, Prasenjit Mitra, and Suhang Wang. 2020. Investigating and Mitigating Degree-Related Biases in Graph Convolutional Networks. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery, 1435–1444.

- [47] Paul Voigt and Axel Von dem Bussche. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide* (1st ed.). Springer International Publishing, Cham.
- [48] Qizhou Wang, Bo Han, Puning Yang, Jianing Zhu, Tongliang Liu, and Masashi Sugiyama. 2025. TOWARDS EFFECTIVE EVALUATIONS AND COMPARISONS FOR LLM UNLEARNING METHODS. In *ICLR*.
- [49] Qizhou Wang, Jin Peng Zhou, Zhanke Zhou, Saebyeol Shin, Bo Han, and Kilian Q. Weinberger. 2025. RETHINKING LLM UNLEARNING OBJECTIVES: A GRADIENT PERSPECTIVE AND GO BEYOND. In *ICLR*.
- [50] Yu Wang, Tong Zhao, Yuying Zhao, Yunchao Liu, Xueqi Cheng, Neil Shah, and Tyler Derr. 2025. TOWARDS EFFECTIVE EVALUATIONS AND COMPARISONS FOR LLM UNLEARNING METHODS. In *ICLR*.
- [51] Jiancan Wu, Yi Yang, Yuchun Qian, Yongduo Sui, Xiang Wang, and Xiangnan He. 2023. GIF: A General Graph Unlearning Strategy via Influence Function. In *Proceedings of the ACM Web Conference 2023 (WWW '23)*. Association for Computing Machinery, New York, NY, USA, 651–661.
- [52] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S. Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2020), 4–24.
- [53] Scholten Yan, Gunnemann Stephan, and Schwinn Leo. 2025. A PROBABILISTIC PERSPECTIVE ON UNLEARNING AND ALIGNMENT FOR LARGE LANGUAGE MODELS. In *ICLR*.
- [54] Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2024. Large Language Model Unlearning. In *ICLR*.
- [55] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2019. Privacy risk in machine learning: Analyzing the connection to overfitting. In *ACM SIGSAC Conference on Computer and Communications Security*.
- [56] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. 2020. XGNN: Towards Model-Level Explanations of Graph Neural Networks. arXiv:2006.02587 [cs.LG]
- [57] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. 2021. On Explainability of Graph Neural Networks via Subgraph Explorations. arXiv:2102.05152 [cs.LG]
- [58] Xiaojian Yuan, Tianyu Pang, Chao Du, Kejiang Chen, Weiming Zhang, and Min Lin. 2025. A CLOSER LOOK AT MACHINE UNLEARNING FOR LARGE LANGUAGE MODELS. In *ICLR*.
- [59] Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. Negative Preference Optimization: From Catastrophic Collapse to Effective Unlearning. In *CoLM*.
- [60] Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. 2025. DO LLMs RECOGNIZE YOUR PREFERENCES? EVALUATING PERSONALIZED PREFERENCE FOLLOWING IN LLMs. In *ICLR*.
- [61] Zhijie Zhu, Lei Fan, Maurice Pagnucco, and Yang Song. 2025. Interpretable Image Classification via Non-parametric Part Prototype Learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 9762–9771.

A Appendices

A.1 Related Work

Graph Unlearning. Retraining[28] refers to train the model from scratch to unlearn specific edges rather than fine-tuning and is inefficient. GraphEraser[3] attempts to achieve graph unlearning by employing graph partitioning and efficient retraining, but it support only node deletion. GraphEditor[8] provides a closed-form solution for linear GNNs to guarantee information deletion, and additional fine-tuning can improve model utility. However, GraphEditor[8] is not designed for graph-structured data, which is only applicable to linear structures. GIF[51] accurately estimates parameter changes by designing influence functions to directly modify the parameters for edge unlearning, but this performance on forget set is poor and can not achieve true unlearning. GNNDelete[5] achieves unlearning by approximating the representation of edges to be forgotten to those that did not exist in the pretrained model, while keeping the neighbors’ representations minimally changed. However, GNNDelete[5] is infeasible to distinguish the representation distance between the forgotten and the retained data, leading to poor robustness for delete ratio. MEGU[25] propose a new mutual evolution paradigm that simultaneously evolves the utility and forget capacities of graph unlearning, which unlearning by gradient ascent with rapid divergence speed. Compared to GNNDelete[5],

UtU[45] only uses the graph after edge deletion for a single inference. However, these models makes the utility vulnerable during the unlearning process due to the rapid divergence speed of gradient ascent, especially MEGU. In this work, we aim to improve the robustness of the model utility to the unlearning process.

Large Language Model Unlearning. Gradient Ascent[54] utilize fine-tuning to minimize correct predictions on the forget set by modifying the cross-entropy loss. NPO[59] adjusts offline DPO[32] to reduce the likelihood of the forget set, avoiding the complexity of learning a reward function like RLHF[9]. SimNPO[12] propose a simple yet effective unlearning optimization framework to remove the reliance on a reference model. To address utility preservation, regularized optimization[31, 36, 58] combines unlearning efficacy with model utility loss, like Gradient Descent loss and KL-Loss. Despite various studies on LLM Unlearning, our study reveals that existing unlearning methods with regularization struggle with handling graph-structure data due to tight coupling between data entities. We propose a simple yet effective solution to improve utility robustness for graph unlearning.

A.2 Overall Performance on All Datasets

Table 6: Comparison results of our model with advanced fine-tuning methods on dataset PubMed.

Model	RT-AUC	FT-AUC	MI Ratio	p_f	p_r	$\frac{p_r}{p_f}$
GIF	0.9643	0.4699	1.05	0.9302	0.9045	0.97
GNNDelete	0.9610	0.9762	1.65	0.5919	0.8692	1.46
UtU	0.9643	0.4585	1.05	0.9288	0.9030	0.97
INPO-S	0.9668	0.9834	1.74	0.5639	0.8717	1.55

Table 7: Comparison results of our model with advanced fine-tuning methods on dataset CS.

Model	RT-AUC	FT-AUC	MI Ratio	p_f	p_r	$\frac{p_r}{p_f}$
GIF	0.9621	0.9129	1.06	0.9240	0.9021	0.97
GNNDelete	0.9515	0.9682	1.68	0.5805	0.8424	1.45
UtU	0.9626	0.5233	1.06	0.9246	0.9027	0.97
INPO-S	0.9525	0.9791	1.80	0.5423	0.8608	1.59

Table 8: Comparison results of our model with advanced fine-tuning methods on dataset OGB-Collab.

Model	RT-AUC	FT-AUC	MI Ratio	p_f	p_r	$\frac{p_r}{p_f}$
GIF	0.9824	0.4837	1.01	0.9665	0.9600	0.99
GNNDelete	0.9850	0.7230	1.45	0.6714	0.8527	1.27
UtU	0.9852	0.5013	1.04	0.9401	0.9340	0.99
INPO-S	0.9827	0.7396	1.56	0.6299	0.8713	1.38

The effectiveness of IMPNN. As shown in Figure 12, IMPNN reduces the adaptive coefficient $S_\theta(x, y)$ leading to a slower divergence speed, thus minimizing the impact on the model utility.

A.3 Hyper-parameters Setting

We list all hyper-parameters setting to reproduce our experiments.

Table 9: Hyper-parameters Setting on All Datasets.

Hyper-parameter	β	epoch	lr	λ_1	NI	λ_3
INPO-S	2	100	1e-3	1	1	0.2

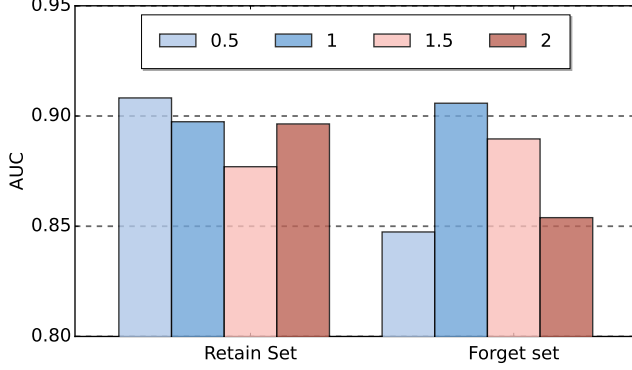


Figure 10: AUC Performance on the retain set and the forget set at different λ_1 .

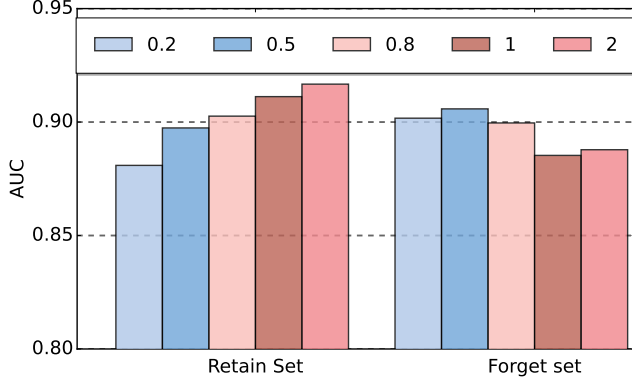


Figure 11: AUC Performance on the retain set and the forget set at different λ_3 .

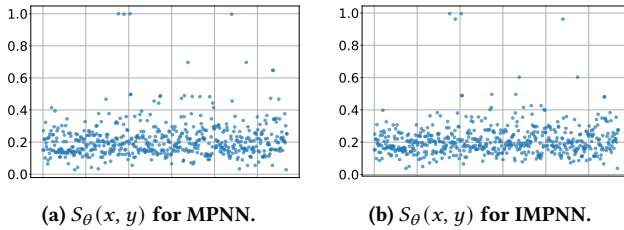


Figure 12: The adaptive coefficient $S_\theta(x, y)$ at epoch 250 for MPNN and IMPNN. A data point represents an edge to be forgotten.

A.4 Proof of Our Model's Effectiveness

In this section, we theoretically prove the effectiveness of the proposed method.

Graph Edge Unlearning is actually to perform separation of edge representations between the forget set and the retain set. The variational form of TV-Divergencen is:

$$\mathcal{D}_{TV}(\mathcal{P}||\mathcal{Q}) = \sup_{f:|f|\leq 1/2} \mathbb{E}_{v\in\mathcal{P}} f(v) - \mathbb{E}_{v\in\mathcal{Q}} f(v), \quad (16)$$

where f an arbitrary function.

The variational form of KL-Divergencen is:

$$\mathcal{D}_{KL}(\mathcal{P}||\mathcal{Q}) = \sup_f \mathbb{E}_{v\in\mathcal{P}} f(v) - \log(\mathbb{E}_{v\in\mathcal{Q}} e^{f(v)}). \quad (17)$$

Let $f = r_\theta(y | x) = \beta \cdot \log[\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}]$, and the simplified loss of NPO is:

$$L_{NPO} = -\mathbb{E}_{(x,y)\in\mathcal{E}_r} r_\theta(y | x) + \mathbb{E}_{(x,y)\in\mathcal{E}_f} r_\theta(y | x). \quad (18)$$

It's obvious that $L_{NPO} = -\mathcal{D}_{TV}$. Therefore, optimizing the NPO loss essentially means increasing the separation of the edge representations between the retain set and the forget set.

The gradient of NPO loss can be expressed as follows:

$$\nabla_\theta \mathcal{L}_{NPO} = -\mathbb{E}_{\mathcal{E}_r} [\nabla_\theta r_\theta(y | x)] + \mathbb{E}_{\mathcal{E}_f} [\nabla_\theta r_\theta(y | x)]. \quad (19)$$

When we enhance the influence of edges on the forget set, the revised loss of NPO similar to KL-Divergence is:

$$L_{INPO} = -\mathbb{E}_{(x,y)\in\mathcal{E}_r} r_\theta(y | x) + \mathbb{E}_{(x,y)\in\mathcal{E}_f} r_\theta(y | x) \cdot e^\xi, \quad (20)$$

where ξ is the influence of edges on the forget set. Now the gradient of NPO is:

$$\nabla_\theta \mathcal{L}_{INPO} = -\mathbb{E}_{\mathcal{E}_r} [\nabla_\theta r_\theta(y | x)] + \mathbb{E}_{\mathcal{E}_f} [\nabla_\theta r_\theta(y | x) \cdot e^\xi]. \quad (21)$$

The above equation means that giving more attention to the forget set reduces the impact on model utility. Therefore, during the unlearning process, the model reduces the prediction probability of forgetting edges, **enlarging the probability difference** between the forget set and the retain set to mitigate topological coupling. Simultaneously, it **preserves the relative influence invariant**.

A.5 Robust for Different Delete Ratio

From the comparison in Table 10-13, we can draw the following conclusions:

- **INPO-S is highly robust to different deletion ratios.** INPO-S consistently outperforms GNNDelete in all metrics of forgetting quality, with **the MI Ratio being on average 7.34% higher**. Notably, as the deletion ratio increases, the MI Ratio of INPO-S even improves.
- **True forgetting maintenance.** As the deletion ratio increases, the p_f of GNNDelete rises with the increase in the deletion ratio, which indicates that true forgetting is not achieved at high deletion ratios. In contrast, the p_f of INPO-S decreases as the deletion ratio increases, suggesting that **higher deletion ratios actually promote effective forgetting**. This is the key difference between the two methods.
- **Good performance maintenance.** As the deletion ratio increases, INPO-S maintains the model's AUC and AP on the Retain set just as well as GNNDelete.

In summary, INPO-S not only achieves significant improvements in forgetting quality and true forgetting metrics but also demonstrates a remarkable enhancement in robustness against deletion

ratios. This illustrates that **Preference Optimization** is an promising approach for effective graph unlearning.

Table 10: The model utility and forget quality on Cora dataset for INPO-S.

Delete Ratio	Retain Set		Forget Set	
	AUC	AP	AUC	AP
0.5	0.9640	0.9638	0.9826	0.9857
1.0	0.9583	0.9575	0.9658	0.9722
1.5	0.9570	0.9545	0.9546	0.9626
2.0	0.9559	0.9538	0.9418	0.9514
2.5	0.9558	0.9534	0.9320	0.9426
5.0	0.9551	0.9532	0.9012	0.9148

Table 11: The Predicted Probability and MI Ratio on Cora dataset for INPO-S.

Delete Ratio	p_f	p_r	$\frac{p_r}{p_f}$	MI Ratio
0.5	0.5473	0.8562	1.56	1.79
1.0	0.5555	0.8109	1.46	1.76
1.5	0.5526	0.7770	1.40	1.77
2.0	0.5514	0.7472	1.36	1.77
2.5	0.5507	0.7324	1.33	1.78
5.0	0.5417	0.6652	1.23	1.80

Table 12: The model utility and forget quality on Cora dataset for GNNDelete.

Delete Ratio	Retain Set		Forget Set	
	AUC	AP	AUC	AP
0.5	0.9609	0.9609	0.9797	0.9838
1.0	0.9626	0.9619	0.9621	0.9692
1.5	0.9632	0.9616	0.9487	0.9565
2.0	0.9643	0.9632	0.9336	0.9412
2.5	0.9646	0.9634	0.9248	0.9321
5.0	0.9669	0.9673	0.8939	0.9000

Table 13: The Predicted Probability and MI Ratio on Cora dataset for GNNDelete.

Delete Ratio	p_f	p_r	$\frac{p_r}{p_f}$	MI Ratio
0.5	0.5595	0.8524	1.52	1.75
1.0	0.5892	0.8198	1.39	1.66
1.5	0.5951	0.7963	1.33	1.64
2.0	0.5994	0.7778	1.29	1.63
2.5	0.6003	0.7682	1.28	1.63
5.0	0.6005	0.7297	1.21	1.63

A.6 Comparison between GNNDelete and INPO

An analysis of the principles of GNNDelete reveals the following two significant issues:

- **Limited forgetting capability.** GNNDelete actually distinguishes edges between the retain set and the forget set by introducing **additional parameters**, which limits its forgetting capability due to the number of parameters. Consequently, as the deletion ratio increases, the quality of forgetting decreases.
- **Structural noise.** The Deleted Edge Consistency(DEC) loss in GNNDelete minimizes the distance between predictions of forget edges and random-chosen node pairs, which would introduce structural noise by encouraging node embeddings in $\mathcal{E}_f$ to reflect random connections rather than forgetting.

The Deleted Edge Consistency loss is:

$$\mathcal{L}_{\text{DEC}}^l = \text{MSE}\left(\left\{[h_u^l; h_v^l] \mid e_{uv} \in \mathcal{E}_f\right\}, \left\{[h_u^l; h_v^l] \mid u, v \in_{\mathcal{R}} V\right\}\right), \quad (22)$$

where MSE refers to Mean-Squared Error, and $\in_{\mathcal{R}}$ means randomly chosen.

We also design experiments from these two perspectives to further validate whether this issue exists in INPO-S as well.

- **Fewer parameters.** We reduce the number of additional parameters to 3/4 and 1/2, and compare the performance of the two models. Implemented using low-rank decomposition.
- **Higher deletion ratio.** We set the deletion ratio to an extremely high value(10%) and compare the performance of the two models.

Table 14: The model utility and forget quality on Cora dataset for INPO-S with fewer parameters(1/2).

Model	Retain Set		Forget Set	
	AUC	AP	AUC	AP
GNNDelete	0.7487	0.7664	0.7636	0.8232
INPO-S	0.7950	0.8029	0.7436	0.7902

Table 15: The Predicted Probability and MI Ratio on Cora dataset for INPO-S with fewer parameters(1/2).

Model	p_f	p_r	$\frac{p_r}{p_f}$	MI Ratio
GNNDelete	0.5018	0.5531	1.10	1.95
INPO-S	0.5091	0.5722	1.12	1.92

From the comparison of Table 14 and 15, we conclude that **the capacity of additional parameters also limits the unlearning ability of INPO-S**, meaning that merely designing from the optimization objective cannot resolve this issue. However, compared to GNNDelete, INPO-S can improve the overall AUC and $\frac{p_r}{p_f}$. This indicates that enlarging the probability gap remains effective even in the case of fewer parameters. The results of Table 16 and 17 also confirms this conclusion.

Table 16: The model utility and forget quality on Cora dataset for GNNDDelete with fewer parameters(3/4).

Model	Retain Set		Forget Set	
	AUC	AP	AUC	AP
GNNDDelete	0.7502	0.7683	0.7696	0.8269
INPO-S	0.7992	0.8085	0.7418	0.7903

Table 17: The Predicted Probability and MI Ratio on Cora dataset for GNNDDelete with fewer parameters(3/4).

Model	p_f	p_r	$\frac{p_r}{p_f}$	MI Ratio
GNNDDelete	0.5018	0.5544	1.10	1.95
INPO-S	0.5096	0.5770	1.13	1.92

The results in Tables 18 and 19 show that INPO-S still maintains strong forgetting ability even at very large delete ratio, particularly in the $\frac{p_r}{p_f}$ metric, outperforming GNNDDelete by 4.95%.

Table 18: The model utility and forget quality on Cora dataset for higher deletion ratio(10%).

Model	Retain Set		Forget Set	
	AUC	AP	AUC	AP
GNNDDelete	0.9683	0.9694	0.8685	0.8720
INPO-S	0.9573	0.9559	0.8756	0.8875

Table 19: The Predicted Probability and MI Ratio on Cora dataset for higher deletion ratio(10%).

Model	p_f	p_r	$\frac{p_r}{p_f}$	MI Ratio
GNNDDelete	0.6041	0.6118	1.01	1.62
INPO-S	0.5326	0.5639	1.06	1.84

Additionally, compared to GNNDDelete, our INPO model also mitigates the strong coupling between data, and this conclusion remains true. This is why it still performs well even at higher deletion ratios. On the contrary, at extremely high deletion ratios, the impact of structural noise becomes more pronounced, causing $\frac{p_r}{p_f}$ to approach 1.

A.7 Mitigate Topological Coupling

INPO enlarges the **probability difference** between the forget set and the retain set to mitigate the topological coupling. By comparing the $\frac{p_r}{p_f}$ metric, it is easy to see that INPO achieves effective topological decoupling. Compared to NPO, INPO improves the $\frac{p_r}{p_f}$ by 22.22% on DBLP dataset and 28.16% on Cora dataset. On the other hand, INPO-S achieves a higher $\frac{p_r}{p_f}$ than GNNDDelete at all deletion ratios. Figure 13 shows **INPO-S achieves large probability difference** and the rate of increase in p_r is significantly greater than that of p_f .

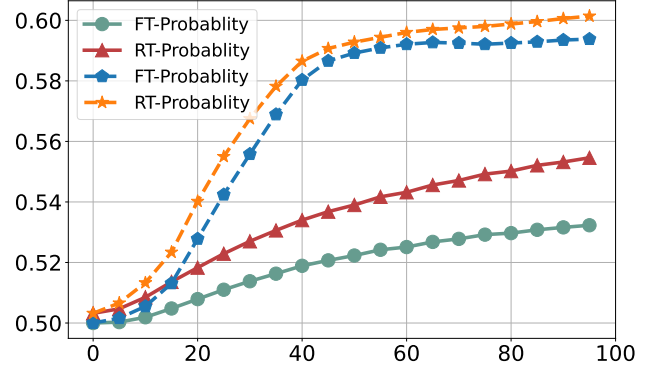


Figure 13: Predicted probability on Cora dataset for delete ratio = 10%. The solid lines denotes INPO-S and the dashed lines represents GNNDDelete.

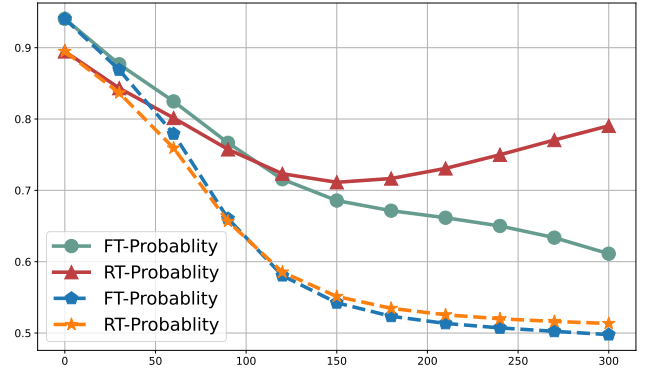


Figure 14: Predicted probability on Cora dataset for delete ratio = 0.5%. The solid lines denotes INPO and the dashed lines represents NPO.

To better illustrate how our method mitigates topological coupling during the training process, we conducted a detailed analysis based on the probability change curves for two scenarios:

- **INPO VS. NPO.** From Figure 14, we can draw three conclusions: (1) INPO effectively enlarges the distance between the retain set and the forget set by maintaining the probability p_r . (2) In NPO, p_r consistently decreases along with p_f throughout the entire training process. (3) In the early stages of INPO, p_r also decreases along with p_f , and only later does a turning point occur. This indicates that INPO only becomes effective when p_f decreases.
- **INPO-S VS. GNNDDelete.** INPO-S uses additional parameters to keep p_f unchanged, thereby achieving unlearning, while continuously improving p_r .

In conclusion, in two scenarios, our **Influence-aware Negative Preference Optimization** achieves effective graph unlearning.

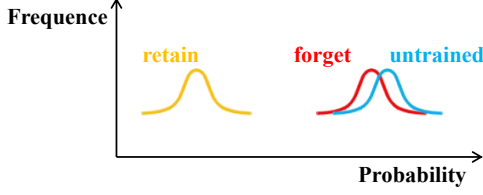


Figure 15: The distribution of ideal unlearned model.

A.8 Why Do We Need Topological Decoupling?

Assume that the prediction probability distribution of a unlearned model on the forget set is denoted as p_f , on the retain set as p_r , and on the untrained set (referring to those edges that **really do not exist**) as p_{ut} . An ideal unlearned model should satisfy the following two conditions:

- **In-distinguishability.** p_f should be as close as possible to p_{ut} ;
- **Separability.** p_f should be as far(separated) as possible from p_r .

We can use Figure 15 to represent an ideal unlearned model. However, although most graph unlearning models currently achieve high AUC and AP on the forget set and retain set, they do not perform well in the two aspects mentioned above, such as GNNDelete and MEGU. We list the In-distinguishability and Separability-related data of models in Table 20. As can be seen from Table 20, directly using NPO for unlearning results in an unlearned model with poor separability (i.e., the unlearning process significantly reduce model utility). This is due to **the strong coupling between entities in the graph data**, and this result can be illustrated using Figure 16.

To address the current challenges faced by these models and NPO, we propose that topological decoupling can better achieve in-distinguishability and separability. Based on Table 20, we can draw the following two conclusions:

- Compared with NPO, INPO enlarges the probability difference between the forget set and the retain set to mitigate the topological coupling, achieves higher separability;
- INPO-S achieves the best in-distinguishability and separability.

Table 20: The In-distinguishability and Separability on Cora. The table shows the mean of the distributions.

Model	p_f	p_{ut}	p_r	In-dist.	Sepa.
NPO	0.5323	0.4726	0.5476	0.0597	0.0153
INPO	0.6070	0.4812	0.8036	0.1258	0.1966
GNNDelete	0.5595	0.4968	0.8524	0.0627	0.2929
INPO-S	0.5473	0.4974	0.8562	0.0499	0.3089

A.9 Model Implementation Details

For NORA, We follow the parameter settings from the original paper, i.e., $k_1 = 1.0$, $k_2 = 0.5$, $k'_2 = 0$, $k_3 = 1$, $\gamma = 8$.

In the implementation of the GCN layer, we adopt the most commonly used MPNN framework. Additionally, We use two GCN

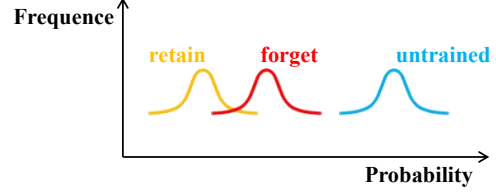


Figure 16: The distribution of current unlearned model.

layers for all graph models. The edge unlearning task is performed on all datasets based on **the link prediction task**.

In the implementation of INPO-S, We discard the Deleted Edge Consistency in GNNDelete and instead adopt the NPO loss and topological entropy loss to forget specific edges. At the same time, we also follow the idea of **adding new parameters** in GNNDelete. We add additional parameters after each GCN layer to adjust the representations of the neighbors around the forgotten edges, with the aim of achieving unlearning.

For fewer parameters settings in Sec. A.6, We adopt the approach of **low-rank decomposition**. The dimension of new additional parameters is 128×128 . We decompose it into two low-rank matrices(A, B) to achieve parameter reduction.

- **1/2 of the total quantity.** The dimension of A is 128×32 , and The dimension of B is 32×128 ;
- **3/4 of the total quantity.** The dimension of A is 128×48 , and The dimension of B is 48×128 .

A.10 Formal Proof of Lemma 3.1

The proof proceeds as follows:

1. According to Eq. 2 in main paper, in the unlearning task, $\nabla_{\theta} \mathcal{L}_{\text{NPO}} \ll \nabla_{\theta} \mathcal{L}_{\text{GA}}$ due to $S_{\theta}(x, y) \ll 1$. Therefore, compared to GA, NPO exhibits very slow parameter updates during unlearning, i.e., **the divergence speed is slow**.

2. The update of θ after unlearning is given by:

$$\theta' = \theta - \eta \cdot \nabla_{\theta} \mathcal{L}_{\text{NPO}}(\theta). \quad (23)$$

3. Considering the change in the loss function on the retain set using a first-order Taylor approximation:

$$\mathcal{L}_{\text{retain}}(\theta) = \mathbb{E}_{(x,y) \sim \mathcal{E}_r} [-\log \pi_{\theta}(y | x)], \quad (24)$$

$$\begin{aligned} \mathcal{L}_{\text{retain}}(\theta') &\approx \mathcal{L}_{\text{retain}}(\theta) + \nabla_{\theta} \mathcal{L}_{\text{retain}}^{\top} \cdot (\theta' - \theta) \\ &= \mathcal{L}_{\text{retain}}(\theta) - \eta \cdot \nabla_{\theta} \mathcal{L}_{\text{retain}}^{\top} \cdot \nabla_{\theta} \mathcal{L}_{\text{NPO}}, \end{aligned} \quad (25)$$

$$\Delta \mathcal{L}_{\text{retain}} \propto \|\nabla_{\theta} \mathcal{L}_{\text{NPO}}\|. \quad (26)$$

Therefore, compared to GA, the performance decrease on the retain set occurs more slowly.

A.11 Formal Proof of Proposition 3.2

The proof proceeds as follows:

1. We assume $h_{\theta}^G(v_i, v_j)$ is the probability of edge (v_i, v_j) on the original graph computed by the model, and let

$$G' = G \setminus \{(v_i, v_j)\}, \quad (27)$$

where the edge (v_i, v_j) is removed from the original graph G . We get the influence of edge (v_i, v_j) :

$$\xi_{ij} = \|h_\theta^G(v_i, v_j) - h_\theta^{G'}(v_i, v_j)\|. \quad (28)$$

2. Given the message passing mechanism of GCN:

$$H^{(l+1)} = \sigma(\hat{A}H^{(l)}W^{(0)}), \quad (29)$$

and we define the edge probability as

$$h_\theta(v_i, v_j) = f(H_{v_i}^{(L)}, H_{v_j}^{(L)}), \quad (30)$$

where $f(\cdot, \cdot)$ is a readout function. Then we get:

$$\frac{\partial h_\theta(v_i, v_j)}{\partial A_{ij}} = \frac{\partial f(H_{v_i}^{(L)}, H_{v_j}^{(L)})}{\partial H_{v_i}^{(L)}} \cdot \frac{\partial H_{v_i}^{(L)}}{\partial A_{ij}} + \frac{\partial f(H_{v_i}^{(L)}, H_{v_j}^{(L)})}{\partial H_{v_j}^{(L)}} \cdot \frac{\partial H_{v_j}^{(L)}}{\partial A_{ij}}. \quad (31)$$

$$\frac{\partial H_{v_i}^{(L+1)}}{\partial A_{ij}} \propto \frac{\partial \hat{A}_{ij}}{\partial A_{ij}} \cdot H_{v_j}^{(L)} W^{(L)} + \sum_{u \in \mathcal{N}(i)} \frac{\partial \hat{A}_{iu}}{\partial A_{iu}} \cdot H_u^{(L)} W^{(L)}. \quad (32)$$

3. If node v_j contributes more significantly to the final representation of node v_i (i.e., $H_{v_j}^{(L)}$ has a larger proportion in $H_{v_i}^{(L+1)}$), then

$\left\| \frac{\partial H_{v_i}^{(L+1)}}{\partial A_{ij}} \right\|$ is larger.

4. According to Step 3, edges with high influence generate larger $\left\| \frac{\partial H_{v_i}^{(L+1)}}{\partial A_{ij}} \right\|$, during the optimization process, leading to **smaller** π_θ by rapid unlearning. Figure 3 in the appendix visualizes this process from the perspective of $S_\theta(x, y)$, and values above 0.4 have been reduced. Smaller $S_\theta(x, y)$ corresponds to smaller $\nabla_\theta \mathcal{L}_{\text{NPO}}$, i.e., slower divergence speed and reduced impact on the retain set.

The most direct goal of ours is to achieve rapid forgetting, while maintaining the performance on the retain set. This is precisely accomplished by amplifying the influence of the edges to be forgotten.