Corrective Unlearning for Graph Neural Networks

MS Thesis Defense

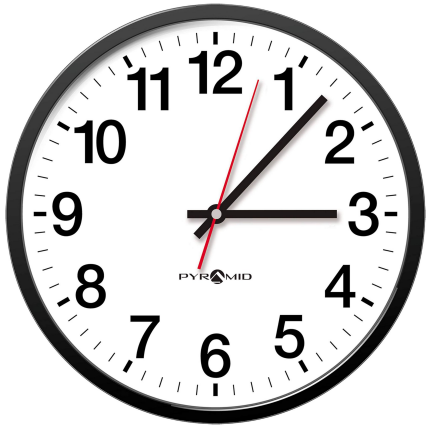
Varshita Kolipaka

Advised by: Prof. Ponnurangam Kumaraguru

List of Related Publications

[P1] Kolipaka, Varshita, Akshit Sinha, Debangan Mishra, Sumit Kumar, Arvindh Arun, Shashwat Goel, and Ponnurangam Kumaraguru. **“A Cognac 🥃 shot to forget bad memories: Corrective Unlearning in GNNs.”**, *International Conference on Machine Learning (ICML), 2025*

[P2] Varshita Kolipaka, Akshit Sinha, Debangan Mishra, Sumit Kumar, Arvindh Arun, Shashwat Goel, Ponnurangam Kumaraguru. **“Sanity Checks For Evaluating Graph Unlearning”**, *CoLLAs, Workshop Track, 2024*



POV

You’re a model developer, and you’ve trained a model with significant time, budget, and compute resources.

And the model generally performs well.

Problem 1: What if a user requests to remove their information from your (very large) trained model?

GDPR Article 17 creates a legal requirement to erase user data upon request.

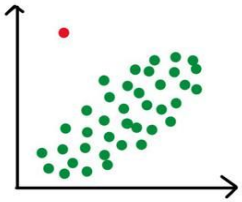
The Technical Problem:

Model parameters θ are a function of the entire training set D.

A single user's influence is distributed throughout the model's weights.

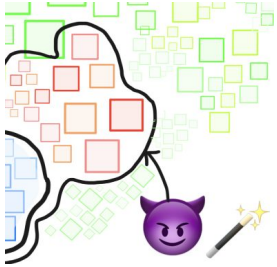
Reconstructed **Original**

Problem 2: What if you find that your (very large) trained model's performance is compromised?

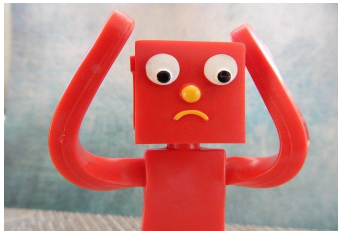


Outliers

Harmful Biases/Unfair predictions



Data Poisoning Attacks



Confusions

We need (efficient) ways to edit models post-training.

Machine Unlearning

Input: Model M trained using learning algorithm A on dataset S_{tr}.

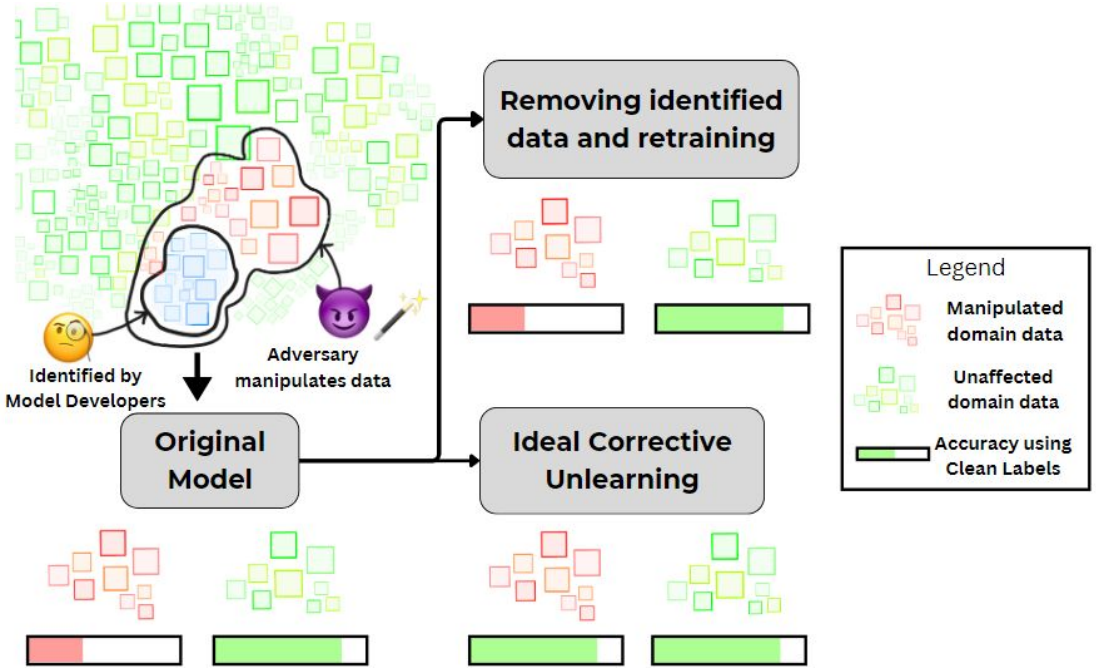
Goal: Unlearning operation U(M, S_{tr}, S_f) for Sf ⊂ S□r that returns a model M_u with the influence of 'S_f' removed, while maintaining other knowledge of M.

Same as 'Deleting' S_f from a trained model.

Source: Goel, S., Prabhu, A., Torr, P., Kumaraguru, P., & Sanyal, A. (2024). Corrective Machine Unlearning. *Transactions on Machine Learning Research* (https://openreview.net/forum?id=v8enu4jP9B)

Special Problem of (Corrective) Unlearning

1. We don't have knowledge of all the manipulated samples, or the exact nature of corruptions and effects.



Corrective Unlearning Goals

Given an identified subset S_f of manipulated samples S_m, achieve the following:

Unlearning: Improve accuracy on samples from affected domain

Utility: Maintain/improve overall test set accuracy

A *Cognac* 🥃 shot to forget bad memories: Corrective Unlearning in GNNs

Varshita Kolipaka*, Akshit Sinha*, Debangan Mishra, Sumit Kumar, Arvindh Arun**, Shashwat Goel** and Ponnurangam Kumaraguru

Paper
(Poster in ICML 2025)

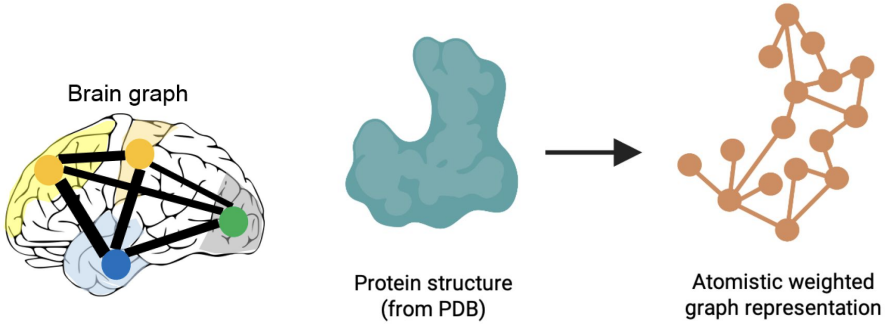
ICML International Conference On Machine Learning

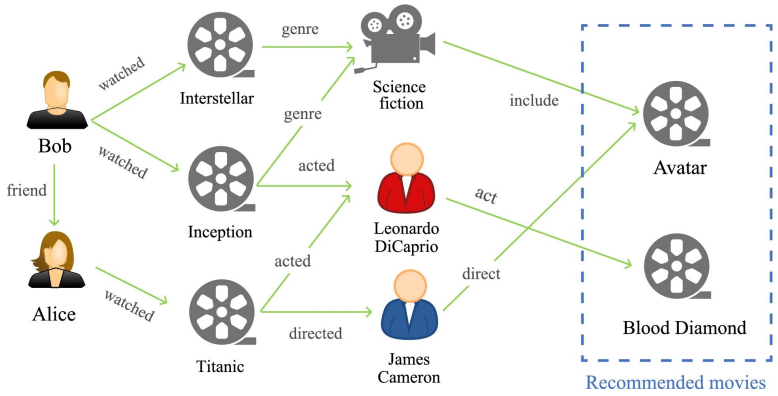
Graph structured data is abundant.

The Entities (Nodes)
Users in a social network, proteins in a biological system, files in a computer network

The Relationships (Edges)

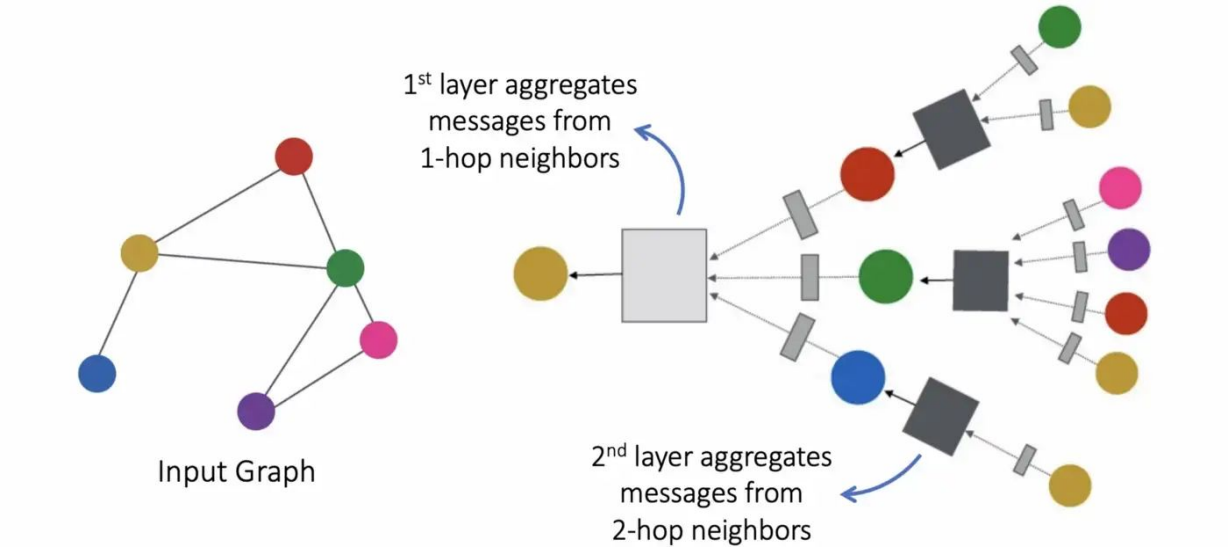
Friendships, followers, physical interactions, dependencies



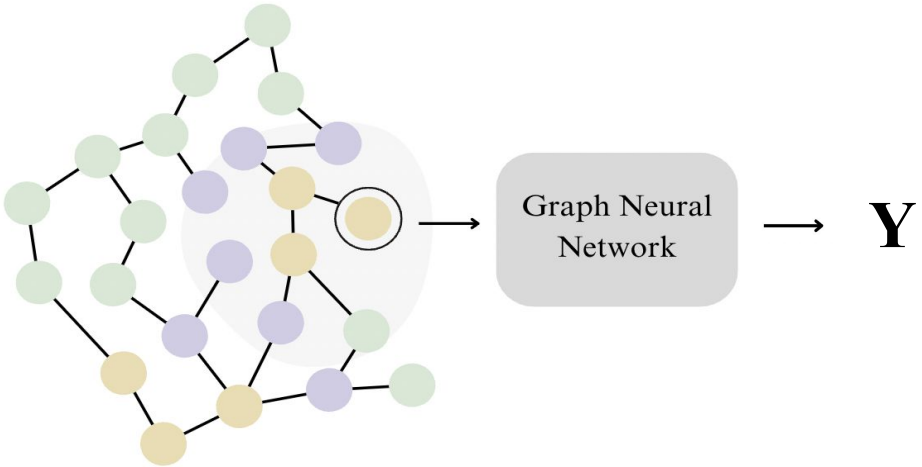


Learning over Graph-Structured Data with GNNs

Graph data is non i.i.d (usefully so), but violates i.i.d. assumption that traditional models rely on.



Why We Studied Graph Unlearning



G(V,E): Graph with nodes **V**, edges **E**
V: (X,Y)
X: Feature Vector
Y: Labels (Not known for test nodes)

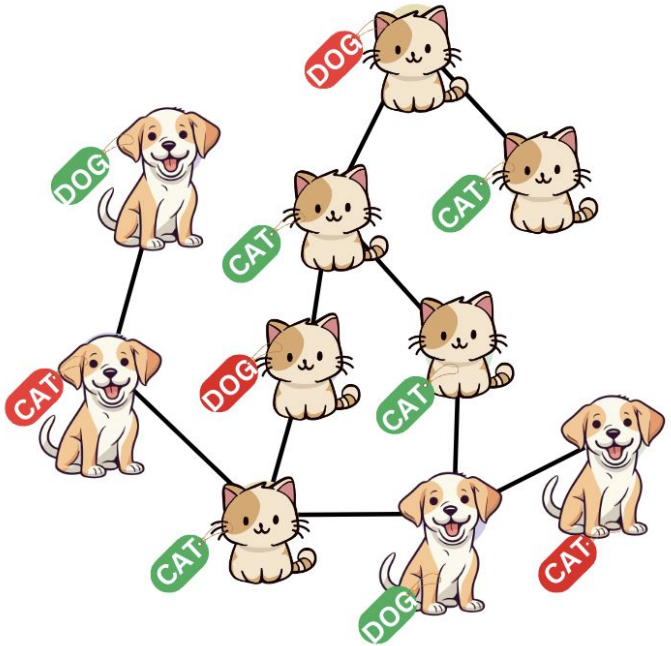
Graph Data is non-i.i.d.

Why it's interesting

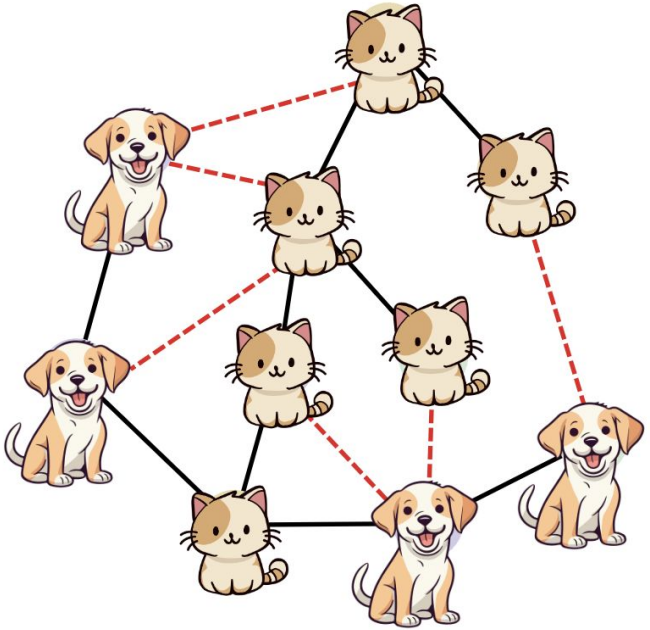
Few adversarial manipulations can greatly affect model performance.

Need to remove the structural effects of manipulations too: i.e., neighbouring regions are affected.

Adversarial Evaluations



Binary Label Flip Attack



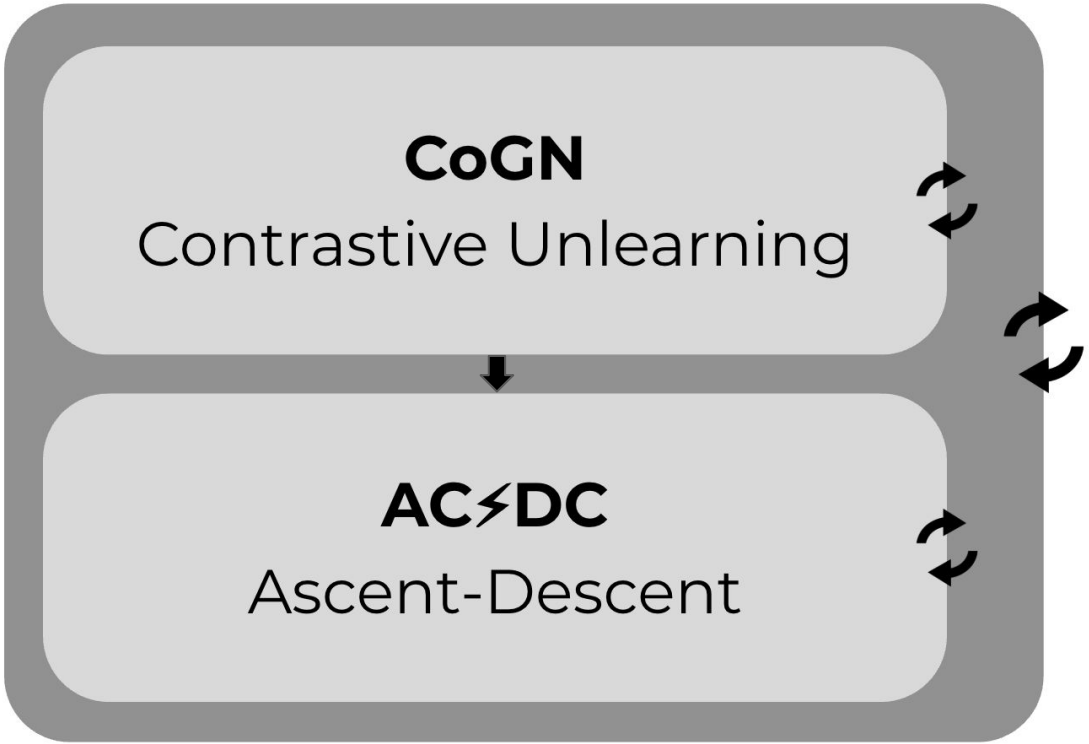
Binary Edge Addition Attack

Baselines

Baselines for Graph Unlearning

- **Reference Models**
  - *Original (Poisoned - Unlearning is done ON this model)*, *Retrain without Sf*, and *Finetune* (continue training without Sf)
  - *Oracle* as an upper bound (trained on unmanipulated data)
- **Graph Unlearning Methods**
  - *GNNDelete:* Deletion operator per GNN layer
  - *GIF:* Closed-form parameter change using the inverse Hessian
  - *MEGU:* Removes influence from highly affected neighborhoods (HIN)
  - *UtU:* Zero-cost edge unlearning by removing edges during inference
- ***i.i.d.* Baseline**
  - *SCRUB:* Teacher–student framework with alternate distillation (adapted for edge unlearning)

Our Method: *Cognac*



Our method consists of two sequential components, which are allowed to repeat for separately tuned number of epochs. The entire algorithm is allowed to repeat for an optimal number of steps.

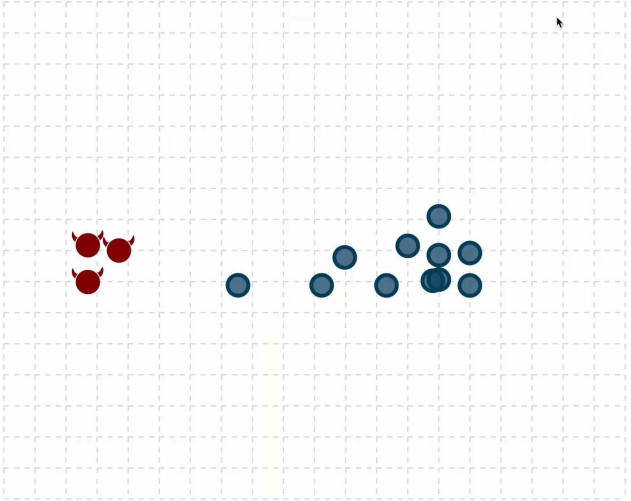
Contrastive Unlearning

$$\mathcal{L}_{v,p,n} = -\log(\sigma(z_v^T z_p)) - \log(\sigma(-z_v^T z_n))$$

Works directly on node representations to push affected nodes toward clean neighbors, away from manipulated nodes

But this can be cheaper!

- **Known**: Manipulated nodes can at most affect n-hop neighbors
- **But n-hop neighborhoods are huge** → expensive to use all nodes as anchors
- **Solution:** Use only 2-10% most affected nodes as anchors (identified via a simple no-learning intervention study)
- **Result:** Same performance, much cheaper computation



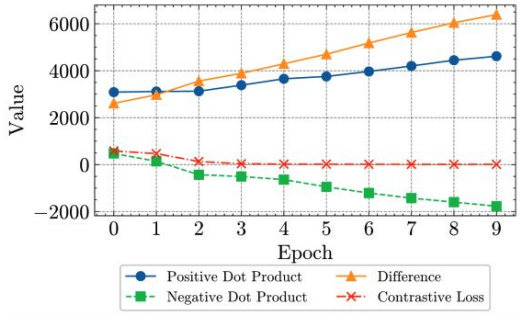
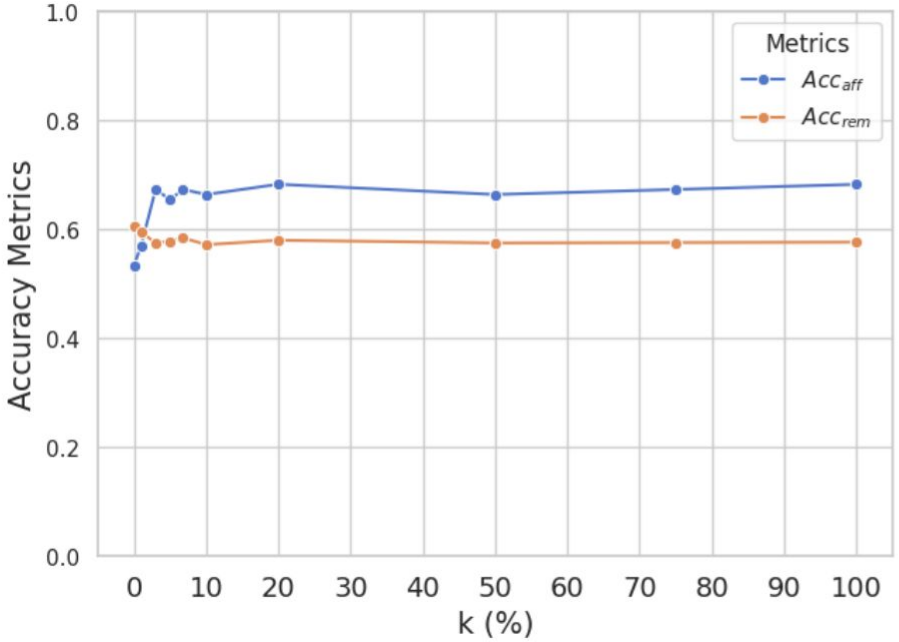
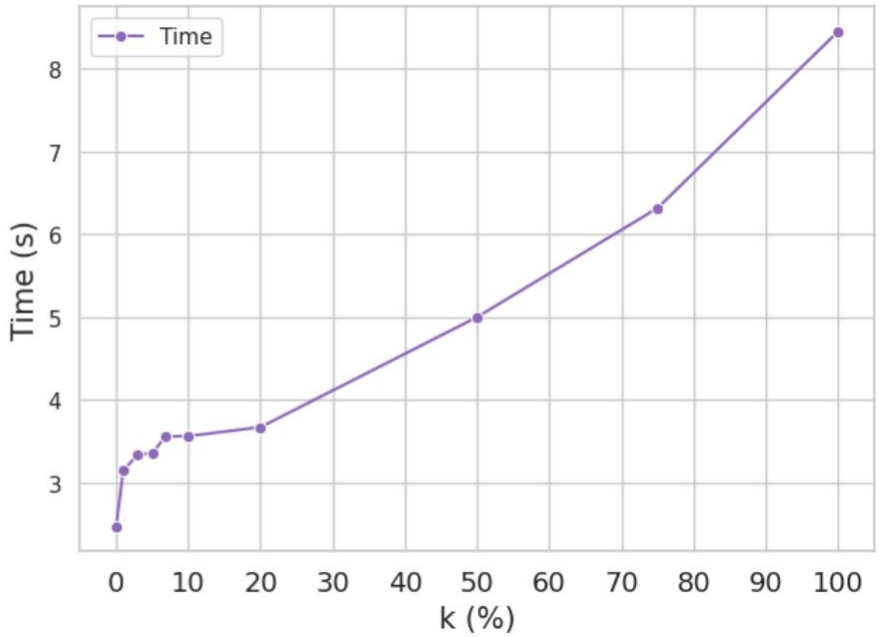


Figure 2. **Empirical convergence of CoGN.** The average positive and negative dot products across samples increase and decrease respectively over epochs, resulting in overall convergence.

Empirical evidence that the selection works

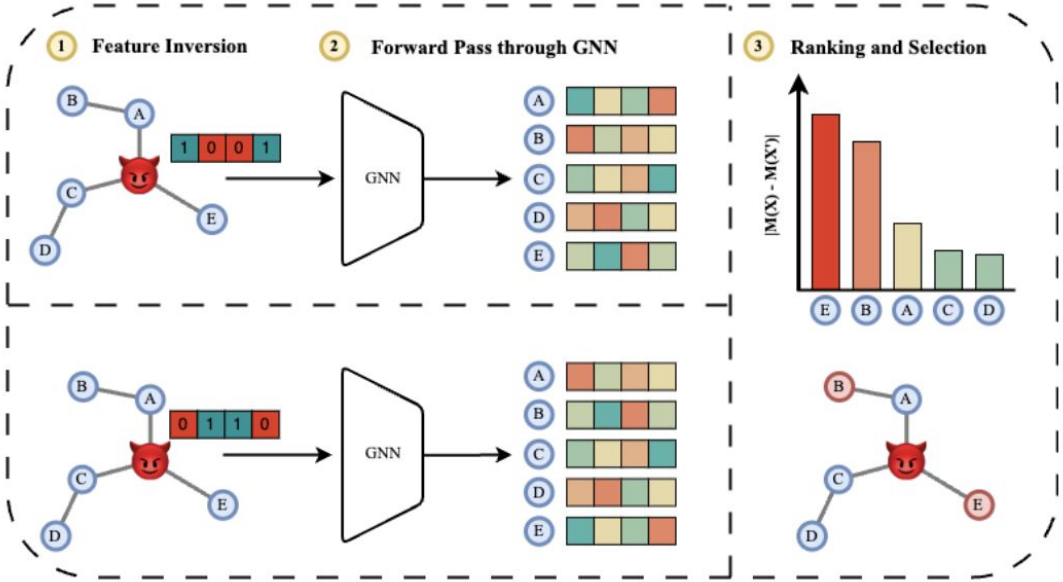




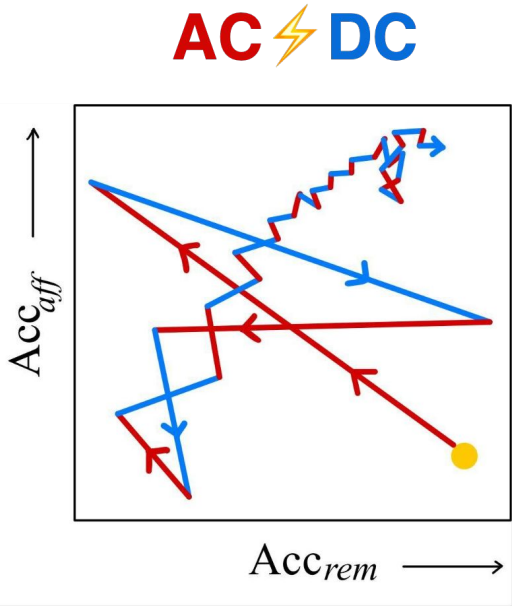
Affected Nodes Identification

"Invert" features of manipulated nodes: $\vec{1} - \mathcal{X}_v \; \forall \, v \in \mathcal{V}_f$

Pick nodes whose logits undergo most change from original: $\Delta\chi = |\mathcal{M}(\chi') - \mathcal{M}(\chi)|$



Ascent/Descent



Perform gradient ascent on the manipulated subset (V_f) and descent on the remaining data (V_r) to undo the effect of wrong labels

$$\mathcal{L}_a = -\mathcal{L}_{\text{task}}(\mathcal{V}_f),\ \mathcal{L}_d = \mathcal{L}_{\text{task}}(\mathcal{V}_r)$$

AC⚡DC: Use separately tuned (typically lower) learning rates for ascent than descent.

Ablations of AC⚡DC

Method	0.25		1.00	
	Acc_{aff}	Acc_{rem}	Acc_{aff}	Acc_{rem}
1 Optimizer 1 LR Alternating Ascent Descent	38.3	56.0	29.0	56.2
1 Optimizer 2 LRs Combined Loss	47.5	56.7	49.6	55.1
AC↯DC	58.8	57.4	54.3	56.1

Do we need both components?

CoGN alone does not improve performance (on the aff. dist.) over the original model

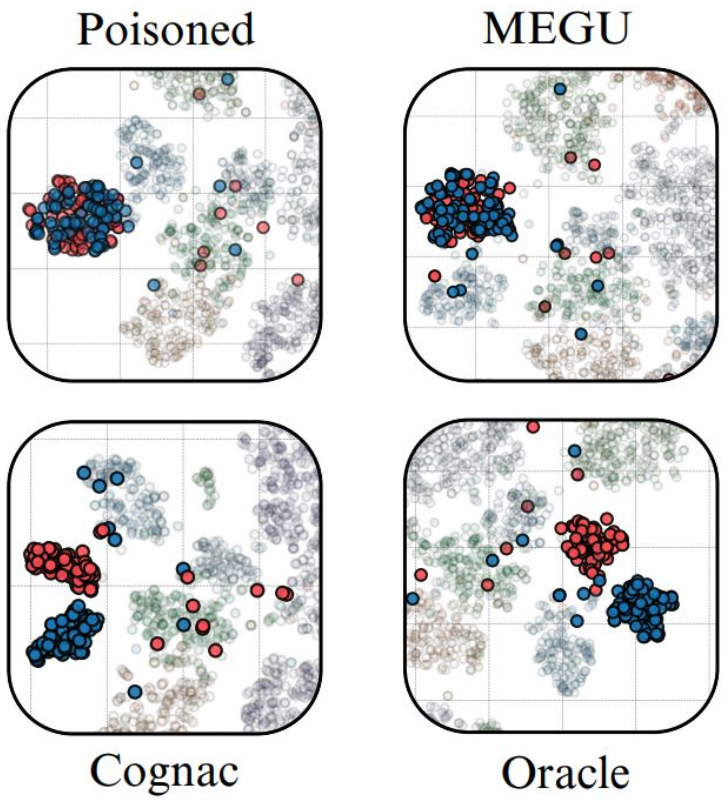
AC⚡DC is a stronger baseline, but Cognac still achieves significant, and consistent gains over it.

METHOD	PHOTOS		CORA		CS	
	Acc_{aff}	Acc_{rem}	Acc_{aff}	Acc_{rem}	Acc_{aff}	Acc_{rem}
ORIGINAL	35.7	94.7	51.4	60.6	44.3	89.8
ORACLE	93.5	94.5	71.1	60.7	90.1	90.0
COGN	34.7	90.7	42.1	60.5	47.2	**89.8**
AC↯DC	76.2	**94.8**	59.9	**61.3**	67.2	86.6
Cognac	**82.9**	92.1	**75.5**	56.6	**79.3**	82.3

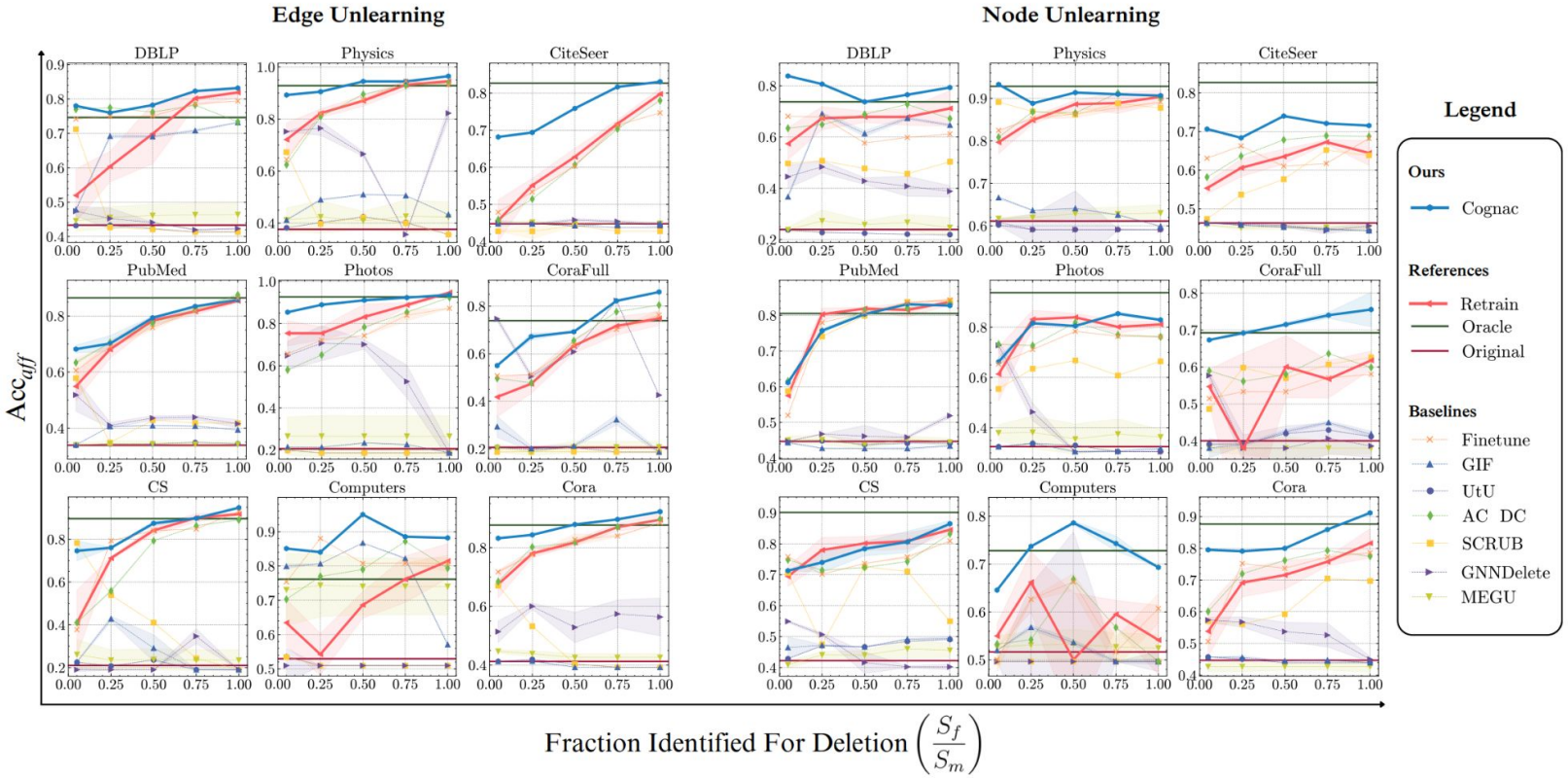
Results

Our method can remove effects of data manipulations even when just **5% of such samples are discovered** (Without compromising model utility).

Most baselines fail to do so even with **100%** discovered.



Results: Recovering Performance on the Affected Distribution



Results: Retaining performance on the ‘clean’ distribution

Method	CS		Cora	
	Edge	Label	Edge	Label
Original	90.6	89.6	61.2	61.4
Oracle	−0.1	+0.5	−0.7	−3.0
Retrain	−1.4	−0.1	−2.4	−5.7
Cognac	−1.4	−0.4	−4.5	−2.9
AC↯DC	−0.7	−0.8	−2.2	−0.8
GNNDelete	−6.0	−1.9	−10.9	−8.4
GIF	−1.6	−0.9	−4.2	−0.6
MEGU	−0.5	−2.8	+0.0	−6.5
UtU	+0.0	+0.0	+0.0	+0.0
SCRUB	−0.9	−0.7	+0.0	−4.8

Some finer grained details

Ablations, and other interesting findings

https://lifelong-ml.cc/Conferences/2024/acceptedpapersandvideos/conf-2024-71

Sanity Checks - Better Baselining and Metrics

Graph unlearning methods have long assumed that i.i.d. Methods would not be useful for this purpose.

In a trigger poisoning experiment, we show that an i.i.d. Method, SCRUB, outperforms one state of the art graph unlearning method.

Takeaway: Despite difference in the image & graph settings, we felt that it’s worth looking to the wider literature for practices, baselines, and metrics

While unlearning ideally assumes nothing about the nature of manipulations, knowledge helps.

Without assumptions, it's common practice in Graph Unlearning literature to unlink the discovered manipulated entities.

But sometimes simply moving the manipulated nodes to the test set can help.

Method	**0.25**		**1.00**	
	Linked	**Unlinked**	**Linked**	**Unlinked**
Oracle	$73.0_{\pm 0.0}$	$73.0_{\pm 0.0}$	$73.0_{\pm 0.0}$	$73.0_{\pm 0.0}$
Original	$42.0_{\pm 0.0}$	$42.0_{\pm 0.0}$	$42.0_{\pm 0.0}$	$42.0_{\pm 0.0}$
Cognac	$\mathbf{64.8_{\pm 0.9}}$	$\mathbf{67.8_{\pm 3.2}}$	$\mathbf{77.2_{\pm 1.0}}$	$\mathbf{69.3_{\pm 1.3}}$
GNNDelete	$35.2_{\pm 2.5}$	$50.2_{\pm 1.9}$	$21.9_{\pm 4.5}$	$30.2_{\pm 5.3}$
MEGU	$40.8_{\pm 1.6}$	$33.4_{\pm 0.4}$	$41.2_{\pm 1.5}$	$32.1_{\pm 1.3}$
SCRUB	$45.7_{\pm 0.0}$	$60.7_{\pm 0.0}$	$41.1_{\pm 0.0}$	$29.0_{\pm 0.0}$
AC↯DC	$61.7_{\pm 0.0}$	$58.8_{\pm 0.0}$	$63.7_{\pm 0.0}$	$54.3_{\pm 0.0}$

This ablation may also shine light on why unlearning at 100% discovery isn't significantly better than at 25% discovery

Problem: Removing more manipulated data paradoxically reduces accuracy on affected classes

Hypotheses for why

Lost Structure Information - Removing manipulated nodes also removes their legitimate connections to real nodes, losing valid structural information

Feature Distribution Shift - Removing edges/nodes changes neighbor feature distributions, making them out-of-distribution for trained GNN layers

Limitations, and Future Work

We evaluate on homophilic datasets, future works can expand this to hetero(philic and geneous) datasets (incl. KGs).

Our method does not guarantee successful unlearning against arbitrary real-world attacks

It would also be interesting to explore the applicability of ***Cognac*** to other domains, such as images.

It would also be practical to use leaner fine tuning methods instead of complete gradient descent on the retain set

Acknowledgements

Prof. Ponnurangam Kumaraguru for the support and freedom to pursue research that is meaningful to me

PreCog Lab collaborators, friends, and other members

Dr. Niket Tandon, Mentor at Microsoft Research

Family and Friends