GraphGuard: Detecting and Counteracting Training Data Misuse in Graph Neural Networks

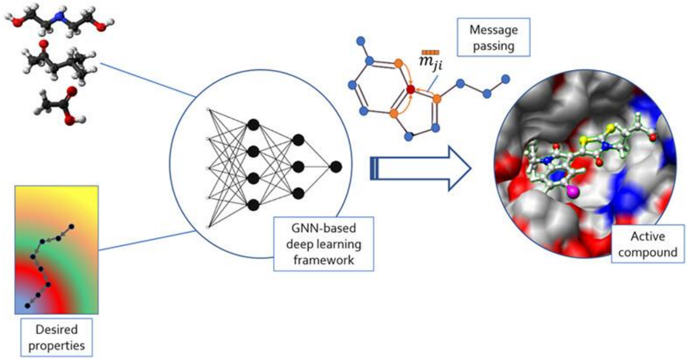
Bang Wu[1,2], **He Zhang**[2], Xiangwen Yang[2], Shuo Wang[1,3], Minhui Xue[1], Shirui Pan[4] and Xingliang Yuan[2]

[1]CSIRO's Data61, [2]Monash University, [3]Shanghai Jiao Tong University, [4]Griffith University

Outline

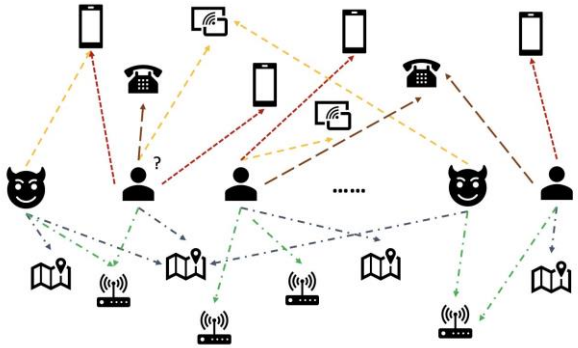
- Background
  - GNNs in MLaaS
  - Graph data misuse
- Our Contribution
- Methodology - GraphGuard
  - Detection
  - Mitigation
- Evaluations

GNNs in Critical Applications



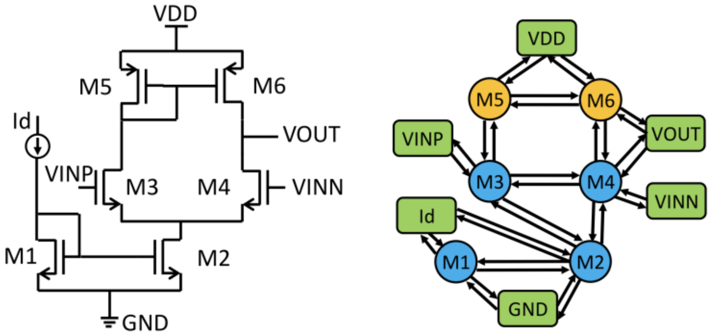
Drug Discovery

[ZYYW+23]



Fraud Detection

[DLSD+20]



Chip Design

[MGYJ+21]

[ZYYW+23] Zhang et al., "Emerging drug interaction prediction enabled by a flow-based graph neural network with biomedical network". *Nature Computational Science* 2023.
[DLSD+20] Dou et al. "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters". In *CIKM* 2020.
[MGYJ+21] Mirhoseini et al., "A graph placement methodology for fast chip design". *Nature* 2021.
Images Source (from left to right):
Abate et al., "Graph neural networks for conditional de novo drug design". *Wiley Interdisciplinary Reviews: Computational Molecular Science* 2023.
https://aws.amazon.com/cn/blogs/machine-learning/build-a-gnn-based-real-time-fraud-detection-solution-using-amazon-sagemaker-amazon-neptune-and-the-deep-graph-library/
Li et al., "A customized graph neural network model for guiding analog IC placement". In *ICCAD* 2020.

GNNs in Machine Learning as a Service (MLaaS)

GNN is increasingly featured on MLaaS platforms:

- **Amazon**: SageMaker Support for DGL
- **Google**: Neo4j & Google Cloud Vertex AI
- **Microsoft**: Azure ML Spektral

AWS Machine Learning Blog

Build a GNN-based real-time fraud detection solution using Amazon SageMaker, Amazon Neptune, and the Deep Graph Library

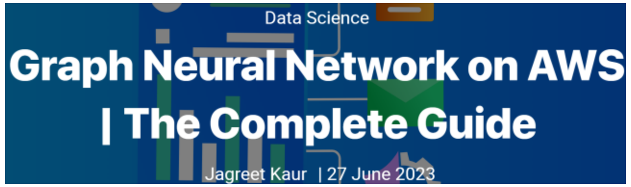
by Jian Zhang, Haozhu Wang, and Mengxin Zhu | on 11 AUG 2022 | in Amazon Neptune, Amazon SageMaker, Artificial Intelligence | Permalink | Comments | Share

Fraudulent activities severely impact many industries, such as e-commerce, social media, and financial services. Frauds could cause a significant loss for businesses and consumers. American consumers reported losing more than $5.8 billion to frauds in 2021, up more than 70% over 2020. Many techniques have been used to detect fraudsters—rule-based filters, anomaly detection, and machine learning (ML) models, to name a few.

amazon | science

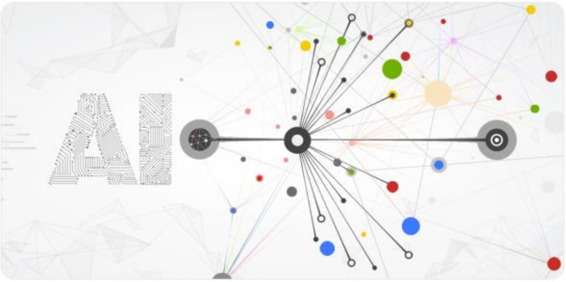
MACHINE LEARNING

How AWS uses graph neural networks to meet customer needs



Use graphs for smarter AI with Neo4j and Google Cloud Vertex AI

January 13, 2022



Ben Lackey

Director, Global Cloud Channel Architecture at Neo4j

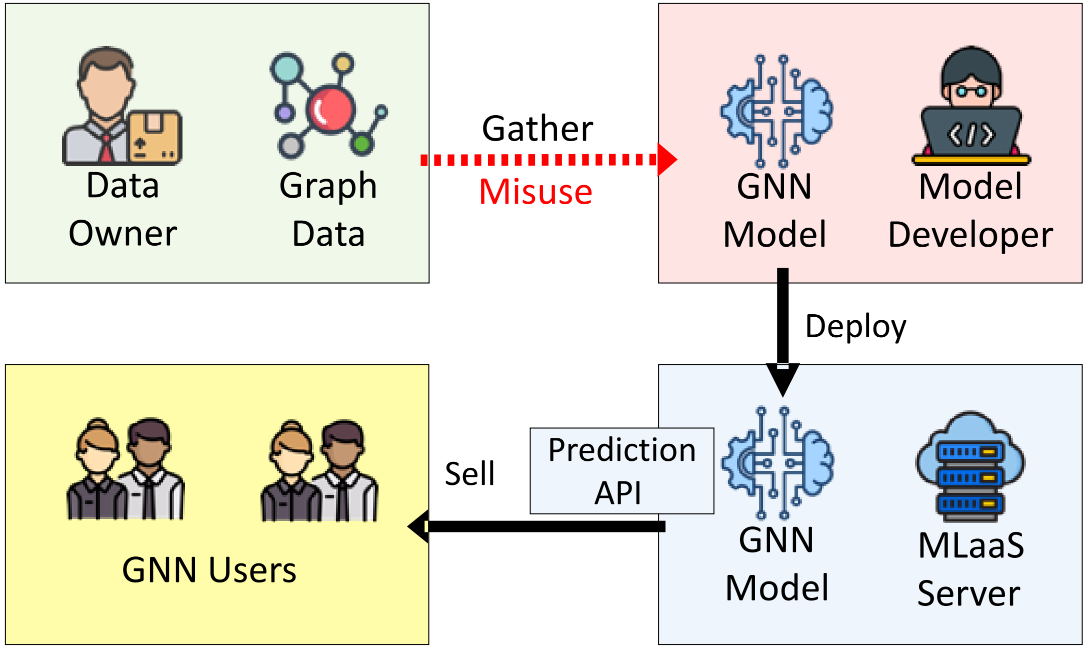
Data Misuse for GNNs in MLaaS

GNN deployment via MLaaS raises data misuse problem

How does misuse data occur in MLaaS?

GNN development via MLaaS

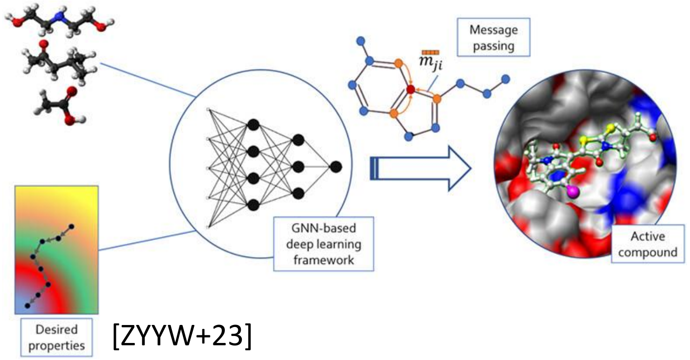
1. **Gather data** for GNN training
2. **Deploy GNNs** via MLaaS
3. **Sell API** to GNN users



Data Misuse in GNNs

GNN deployment via MLaaS raises data misuse problem

Graph can be illegally/wrongfully collected for GNN training!

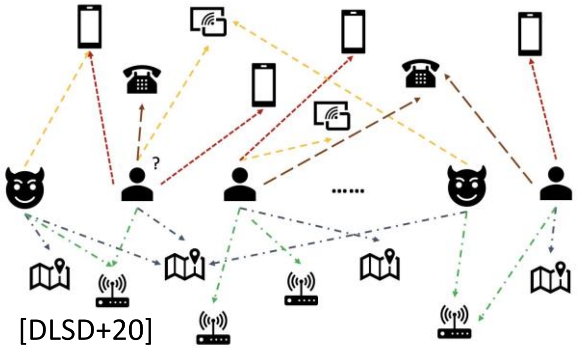


[ZYYW+23]

Drug Discovery

Mislead GNN prediction

Data error leads to incorrect predictions

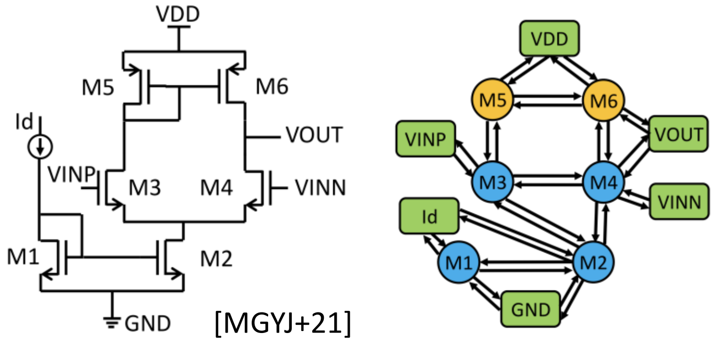


[DLSD+20]

Fraud Detection

Leak sensitive data

Transaction records are private



[MGYJ+21]

Chip Design

Compromise IP of data owners

Chip floorplan needs intellectual effort

How to deal with data misuse?

- **Detection**--*Membership Inference*
  - Identify if specific graph have been used without authorisation
    - *Stealing Links [HJBG+]*
    - *Node-Level Membership Inference [HWWB+21]*
    - *Graph-level Membership Inference [WYPY21]*
- **Mitigation--***Machine Unlearning*
  - Make GNN model forget about specific misused graph data
    - *GraphEraser [CZWB+22]*
    - *GNNDelete [CDHA+23]*

[HJBG+21] [HWWB+21] He, Xinlei, et al. "Node-level membership inference attacks against graph neural networks." *arXiv* 2021.
[WYPY21] Wu, Bang, et al. "Adapting membership inference attacks to GNN for graph classification: Approaches and implications." *ICDM* 2021.
[CZWB+22] Chen, Min, et al. "Graph unlearning." *CCS* 2022.
[CDHA+23] Cheng, Jiali, et al. "GNNDelete: A General Strategy for Unlearning in Graph Neural Networks." *ICLR 2023*.

Requirements of Mitigating Data Misuse in MLaaS

- Task Requirements

  R1 - **Misuse Detection** - Detect the data misused GNNs

  R2 - **Misuse Mitigation** - Remove the misused data's impact

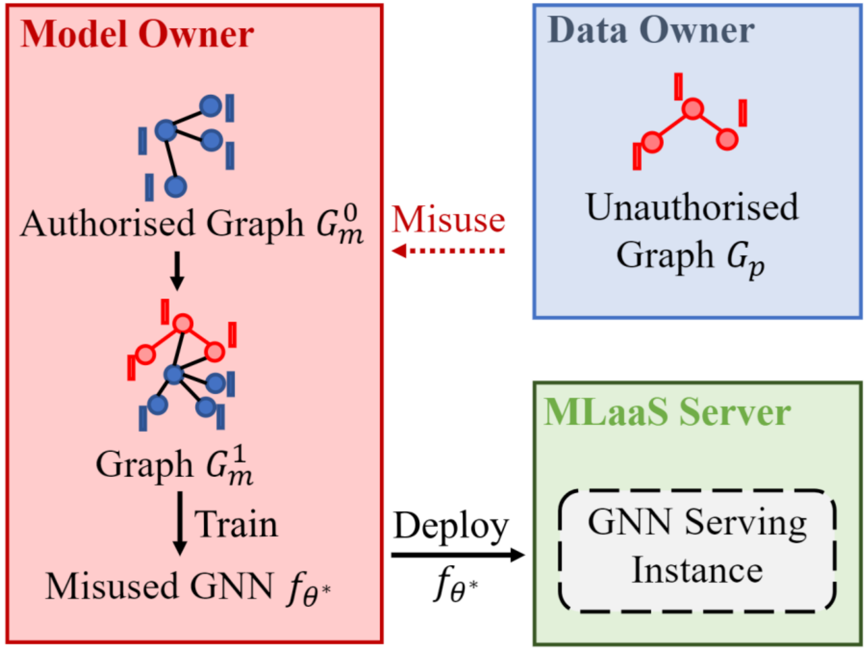
- (MLaaS) Setting Requirements

  R3 - **Data Privatisation** - Keep sensitive data locally

  R4 - **GNN Model Agnostic** - No assumption on GNN training/model architecture

Our Design -- *GraphGuard*

- Identify if G_p is used in f_{θ^*} training (**R1**)
  - Membership inference
- Eliminate the impact of G_p on f_{θ^*} (**R2**)
  - Unlearning
- Do not leverage the graph structure (**R3**)
- Utilise only standard APIs in MLaaS (**R4**)

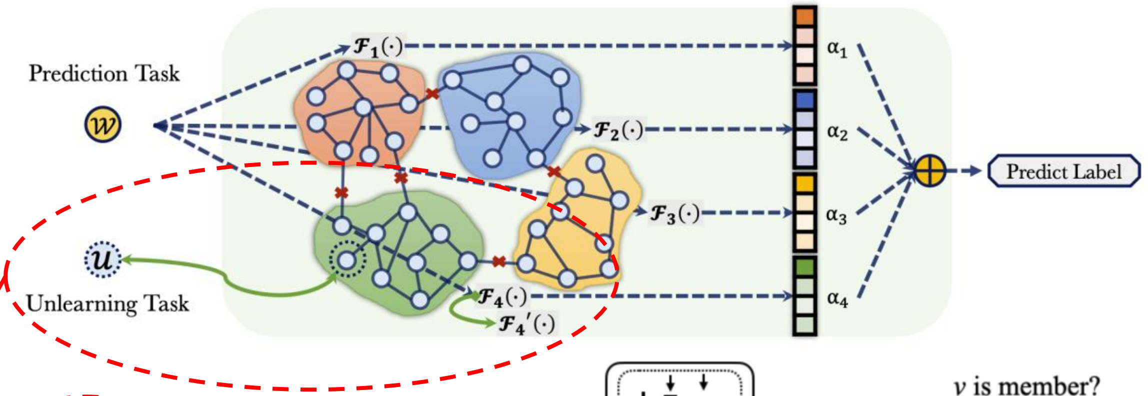


Prior Work

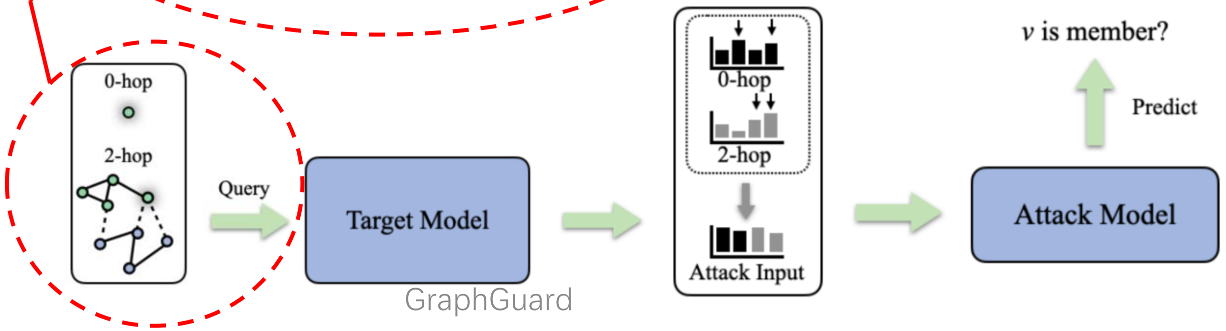
Existing works are less compatible for MLaaS:

- **Assume server to access exact training samples**
- Require additional functions in GNN architecture or training process

Querying the exact training graph to server.



[HWWB+21]

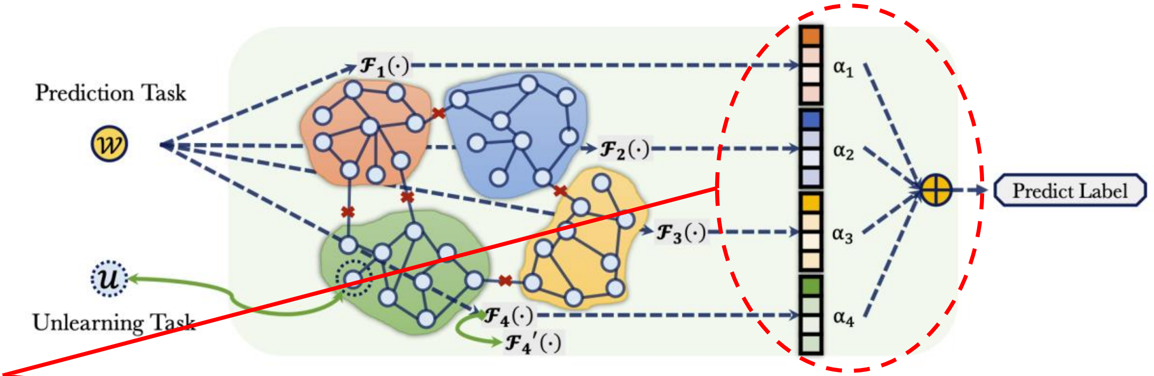


[CZWB+22]

Prior Work

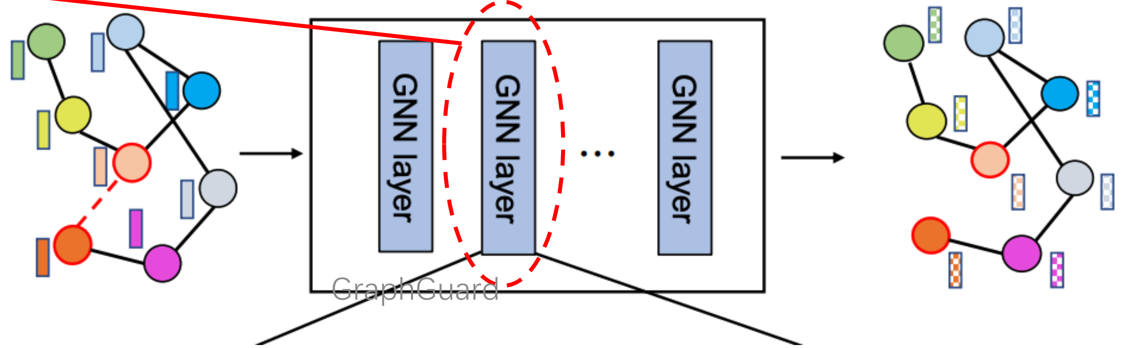
Existing works are less compatible for MLaaS:

- Assume server to access exact training samples
- **Require additional functions in GNN architecture or training process**



[HWWB+21]

Additional functions in GNNs.



[CDHA+23]

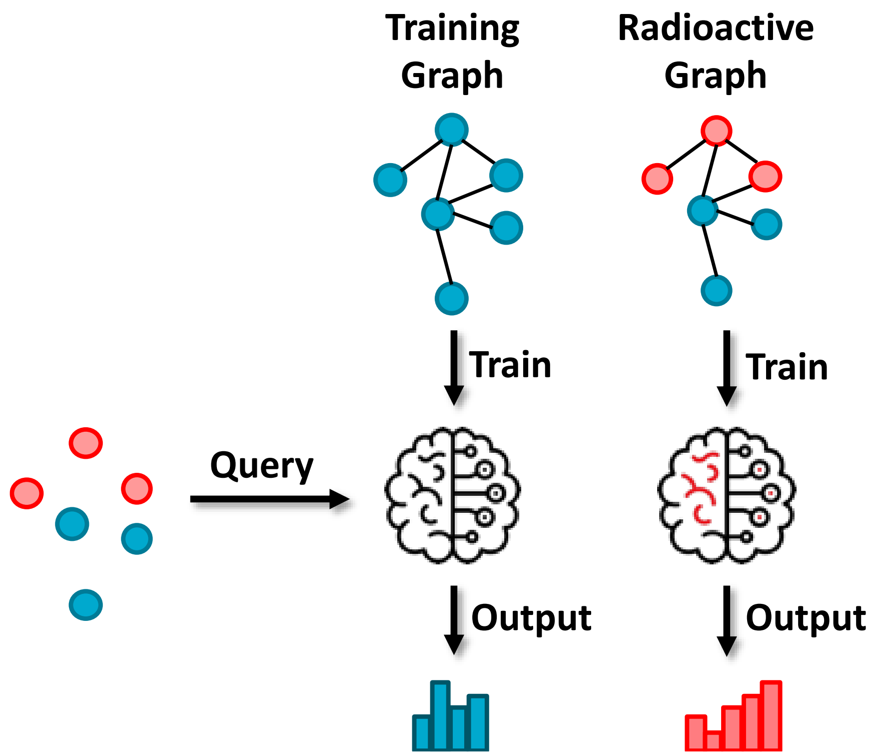
GraphGuard - Detection

- Detection goal

  Detect data misuse (**R1**) via API (**R4**) without the graph structure (**R3**)

- How to perform membership inference without the graph structure?
  - Prior study: proactive MIA. [SDSJ20]
  - Our design: radioactive graph

GNNs trained on them react differently for specific node attribute queries



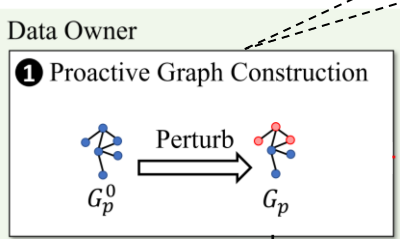
[SDSJ20] Sablayrolles, A., et.al. Radioactive data: tracing through training. ICML 2020.

GraphGuard - Detection

Pipeline:

1. Revise node attributes from G_b to G_p before publishing graph

$$\begin{aligned} &\max_{G_p} d(\mathcal{A}(f_{\theta_1^*}(\hat{G}_p)), \mathcal{A}(f_{\theta_0^*}(\hat{G}_p))), \\ &s.t.\ \theta_1^* = \arg\min_{\theta} L(f_\theta(G_m^1)), \\ &\qquad \theta_0^* = \arg\min_{\theta} L(f_\theta(G_m^0)), \end{aligned}$$

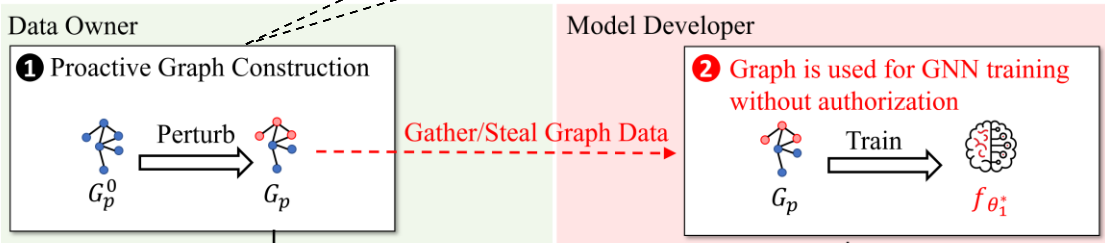


GraphGuard - Detection

Pipeline:

1. Revise node attributes from G_b to G_p before publishing graph
2. Data misuse during training
3. GNN being deployed

$$\max_{G_p} d(\mathcal{A}(f_{\theta_1^*}(\hat{G}_p)), \mathcal{A}(f_{\theta_0^*}(\hat{G}_p))),$$
$$s.t.\ \theta_1^* = \arg\min_{\theta} L(f_\theta(G_m^1)),$$
$$\theta_0^* = \arg\min_{\theta} L(f_\theta(G_m^0)),$$

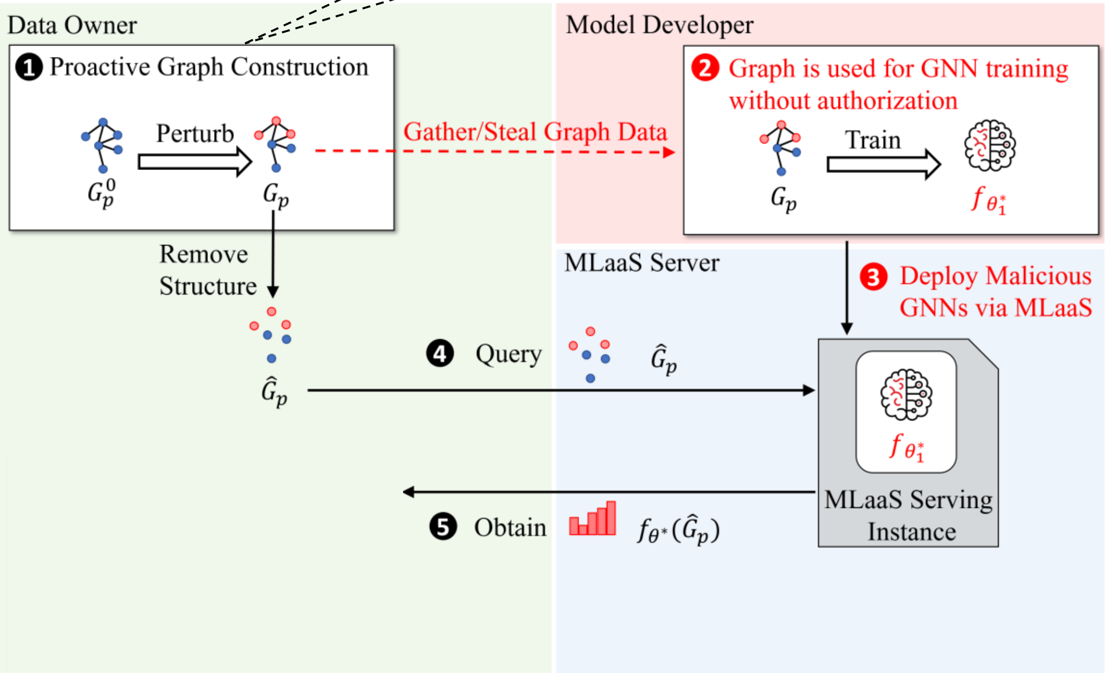


GraphGuard - Detection

Pipeline:

1. Revise node attributes from G_b to G_p before publishing graph
2. Data misuse during training
3. GNN being deployed
4. Query graph $\hat{G}_p$ with node attributes only (without structure)
5. Obtain predictions $f_{\theta^*}(\hat{G}_p)$

$$\max_{G_p} d(\mathcal{A}(f_{\theta_1^*}(\hat{G}_p)), \mathcal{A}(f_{\theta_0^*}(\hat{G}_p))),$$
$$s.t.\ \theta_1^* = \arg\min_{\theta} L(f_\theta(G_m^1)),$$
$$\theta_0^* = \arg\min_{\theta} L(f_\theta(G_m^0)),$$

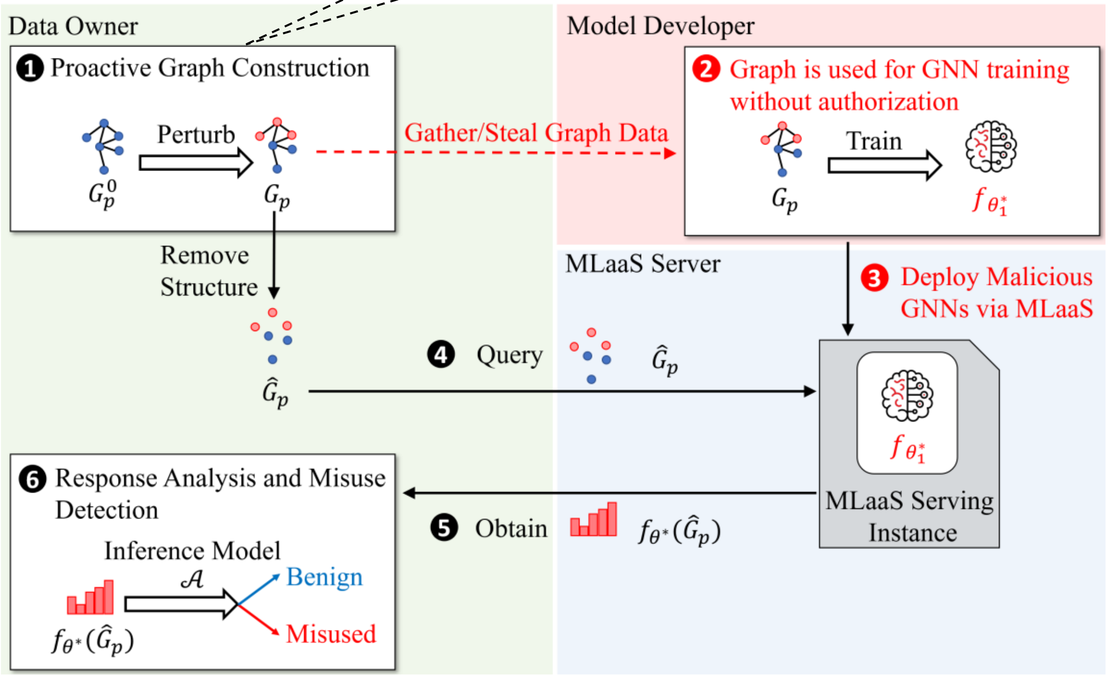


GraphGuard - Detection

Pipeline:

1. Revise node attributes from G_b to G_p before publishing graph
2. Data misuse during training
3. GNN being deployed
4. Query graph $\hat{G}_p$ with node attributes only (without structure)
5. Obtain predictions $f_{\theta^*}(\hat{G}_p)$
6. Membership inference $\hat{\mathcal{A}}$ (difference in output distributions)

$$\boxed{\max_{G_p} d(\mathcal{A}(f_{\theta_1^*}(\hat{G}_p)), \mathcal{A}(f_{\theta_0^*}(\hat{G}_p))),}$$
$$s.t.\ \theta_1^* = \arg\min_{\theta} L(f_\theta(G_m^1)),$$
$$\theta_0^* = \arg\min_{\theta} L(f_\theta(G_m^0)),$$



GraphGuard - Mitigation

- Mitigation goal

  Perform unlearning (R2) by fine-tuning the target GNNs (R4) without utilising the exact graph structure (R3)

GraphGuard - Mitigation

- Mitigation goal

  Perform unlearning (**R2**) by fine-tuning the target GNNs (**R4**) without utilising the exact graph structure (**R3**)

- Design intuitions
  - Well-generalised GNNs **do not learn the exact graph structure**
  - Unlearning a subgraph **does not rely on the exact sub-graph structure**

GraphGuard - Mitigation

- Mitigation goal

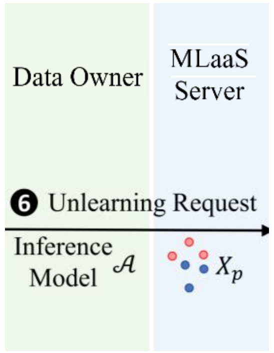
  Perform unlearning (**R2**) by fine-tuning the target GNNs (**R4**) without utilising the exact graph structure (**R3**)

- Design intuitions
  - Well-generalised GNNs **do not learn the exact graph structure**
  - Unlearning a subgraph **does not rely on the exact sub-graph structure**

- Our design
  - Leverage MIA for **graph synthesis**
  - Use synthetic graph for unlearning

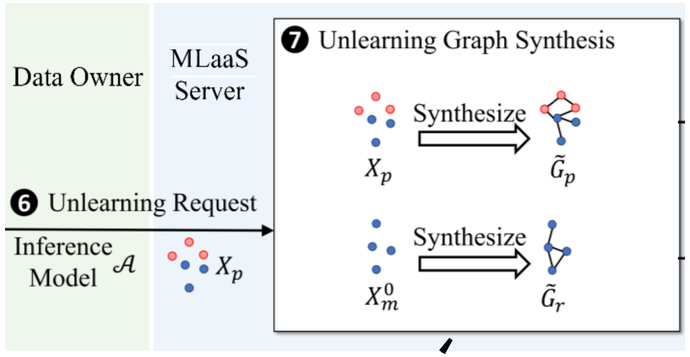
GraphGuard - Mitigation

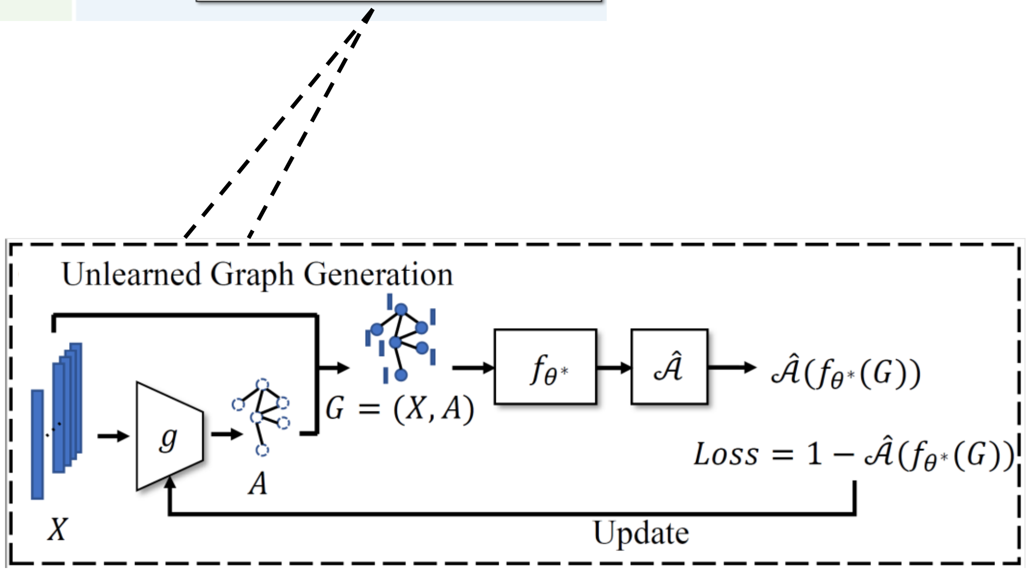
6. MLaaS receives an unlearning request



GraphGuard - Mitigation

6. MLaaS receives an unlearning request
7. (1) Data Gathering
   X_p, $\hat{\mathcal{A}}$ from the data owner
   X_m^0 from the model owner
7. (2) Graph Synthesise
   Unlearning graph $\tilde{G}_p$ by X_p, f_{θ^*} and $\hat{\mathcal{A}}$
   Remaining graph $\tilde{G}_r$ by X_m^0 , f_{θ^*} and $\hat{\mathcal{A}}$





GraphGuard - Mitigation

6. MLaaS receives an unlearning request
7. (1) Data Gathering

   X_p, $\hat{\mathcal{A}}$ from the data owner

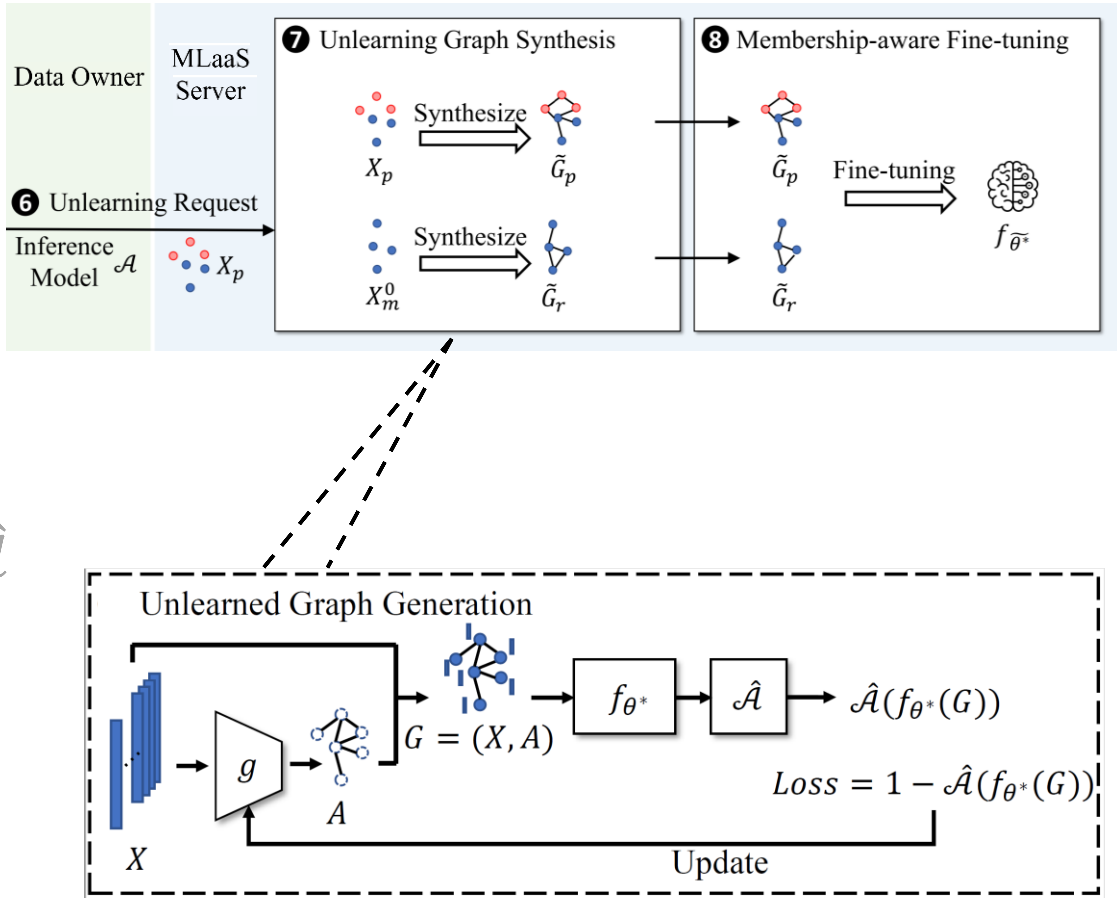
   X_m^0 from the model owner
7. (2) Graph Synthesise

   Unlearning graph $\tilde{G}_p$ by X_p, f_{θ^*} and $\hat{\mathcal{A}}$

   Remaining graph $\tilde{G}_r$ by X_m^0 , f_{θ^*} and $\hat{\mathcal{A}}$
8. Fine-tuning f_{θ^*} :

   Increase loss on $\tilde{G}_p$

   Decrease loss on $\tilde{G}_r$



Evaluations - Detection

	GCN			GraphSage			GAT			GIN		
	Baseline	Ours	Δ	Baseline	Ours	Δ	Baseline	Ours	Δ	Baseline	Ours	Δ
Cora	0.874	0.999	↑**0.125**	0.864	0.999	↑**0.135**	0.927	1.0	↑**0.073**	0.857	1.0	↑**0.143**
Citeseer	0.711	0.999	↑**0.288**	0.822	1.0	↑**0.178**	0.723	0.999	↑**0.276**	0.767	1.0	↑**0.233**
Pubmed	0.906	1.0	↑**0.094**	0.902	1.0	↑**0.098**	1.0	1.0	**0**	0.932	1.0	↑**0.068**
Flickr	1.0	1.0	**0**	0.994	1.0	↑**0.006**	0.996	1.0	↑**0.004**	0.998	1.0	↑**0.002**

Metric - AUC

Observations

- Our design achieves higher detection rates
- Baseline MIA only satisfied R1-Detectable & R4-Model Agnostic

Evaluations - Mitigation

- **Effectiveness** - MIA ASR before/after unlearning

	GCN			GraphSage			GAT			GIN		
	Before	After	Δ	Before	After	Δ	Before	After	Δ	Before	After	Δ
Cora	86.9	51.8	↓ **35.1**	83.3	54.5	↓ **28.8**	85.6	47.5	↓ **38.1**	91.7	47.9	↓ **43.8**
Citeseer	91.3	68.7	↓ **22.6**	81.2	56.1	↓ **25.1**	61.4	60.3	↓ **1.10**	86.2	46.2	↓ **40.0**
Pubmed	93.6	49.2	↓ **44.4**	85.7	53.2	↓ **32.5**	82.4	49.7	↓ **32.7**	84.1	47.6	↓ **36.5**

- **Utility -** Model ACC before/after unlearning

	GCN			GraphSage			GAT			GIN		
	R	U	Δ	R	U	Δ	R	U	Δ	R	U	Δ
Cora	75.7	74.3	↓ **1.2**	67.4	66.5	↓ **0.9**	83.1	81.5	↓ **1.6**	86.4	85.1	↓ **1.3**
Citeseer	81.1	80.0	↓ **1.1**	70.0	68.7	↓ **1.3**	82.2	80.1	↓ **2.1**	79.5	78.9	↓ **0.6**
Pubmed	81.8	79.8	↓ **2.0**	82.5	80.3	↓ **2.2**	83.6	81.3	↓ **2.3**	83.6	82.8	↓ **0.8**

Evaluations - Mitigation

- **Efficiency** - Time cost of retraining/our unlearning

	GCN			GraphSage		
	R	Ours	Times(↑)	R	Ours	Times(↑)
Cora	3.615	0.725	≈**4.99**	4.188	0.643	≈**6.51**
Citeseer	1.746	0.375	≈**4.66**	2.023	0.333	≈**6.08**
Pubmed	4.201	3.043	≈**1.38**	4.865	2.670	≈**1.82**

	GAT			GIN		
	R	Ours	Times(↑)	R	Ours	Times(↑)
Cora	3.600	0.720	≈**5.0**	4.26	1.225	≈**3.48**
Citeseer	1.737	0.375	≈**4.63**	2.058	0.613	≈**3.56**
Pubmed	4.190	3.017	≈**1.39**	4.968	5.124	≈**0.97**

Conclusion

- Definition of New Problem
  - We define the graph misuse in MLaaS-deployed GNNs
- Requirement formulation
  - **Task Requirements**: (R1) detectable, (R2) remedial
  - **(MLaaS) Setting Requirements**: (R3) data privatisation, (R4) model agnostic
- An Integrated Pipeline
  - **Radioactive data** driven detection technique
  - Unlearning methodology w/o confidential graph structure

Q&A

- Code: https://github.com/GraphGuard/GraphGuard-Proactive.
- Email: bang.wu@data61.csiro.au, he.zhang1@monash.edu.

Code

Paper