UNIVERSITY OF WATERLOO

CrySP
Cryptography, Security, and Privacy Research Group

GrOVe: Ownership Verification of Graph NNs using Embeddings

Asim Waheed, Vasisht Duddu, N. Asokan

asim.waheed@uwaterloo.ca, vasisht.duddu@uwaterloo.ca , asokan@acm.org

IEEE Symposium on Security and Privacy, 2024

Introduction

Graph NNs (GNNs) are the state-of-the-art for real-world graph-based applications

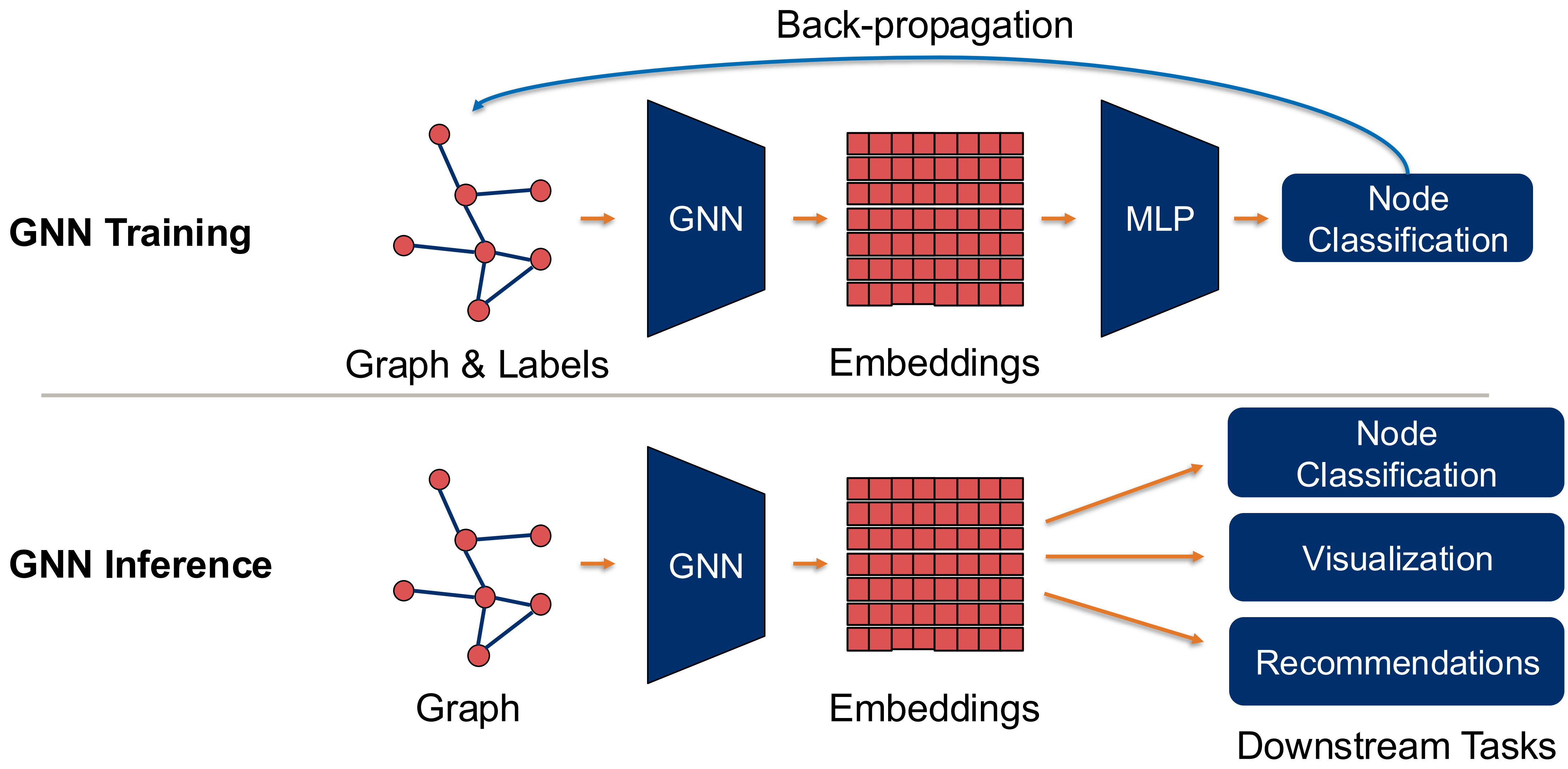
- GNNs require significant resources and data to train

Prior work[1] has shown model extraction is possible on GNNs

- Need for ownership demonstration

[1] Shen et al. *Model Stealing Attacks Against Inductive Graph Neural Networks,* IEEE SP, (2022).

Background: GNN Training and Inference



(How) can we design an ownership verification technique for GNNs?

Model Extraction Attacks on GNNs

Practical Setting: Model extraction for inductive GNNs[1]

Two Attacks

- **Type 1:** Adversary has adjacency matrix and directly trains surrogate model
- **Type 2:** Adversary estimates adjacency matrix before training surrogate model

High accuracy on primary task

High fidelity between target and surrogate model

[1] Shen et al. *Model Stealing Attacks Against Inductive Graph Neural Networks,* IEEE SP, (2022).

Ownership Verification: Desiderata

Effective

Differentiate between surrogate and independent models

Robust

Resists attempts at circumventing ownership verification (compression, fine-tuning)

Efficient

Reasonable computational overhead

Accurate

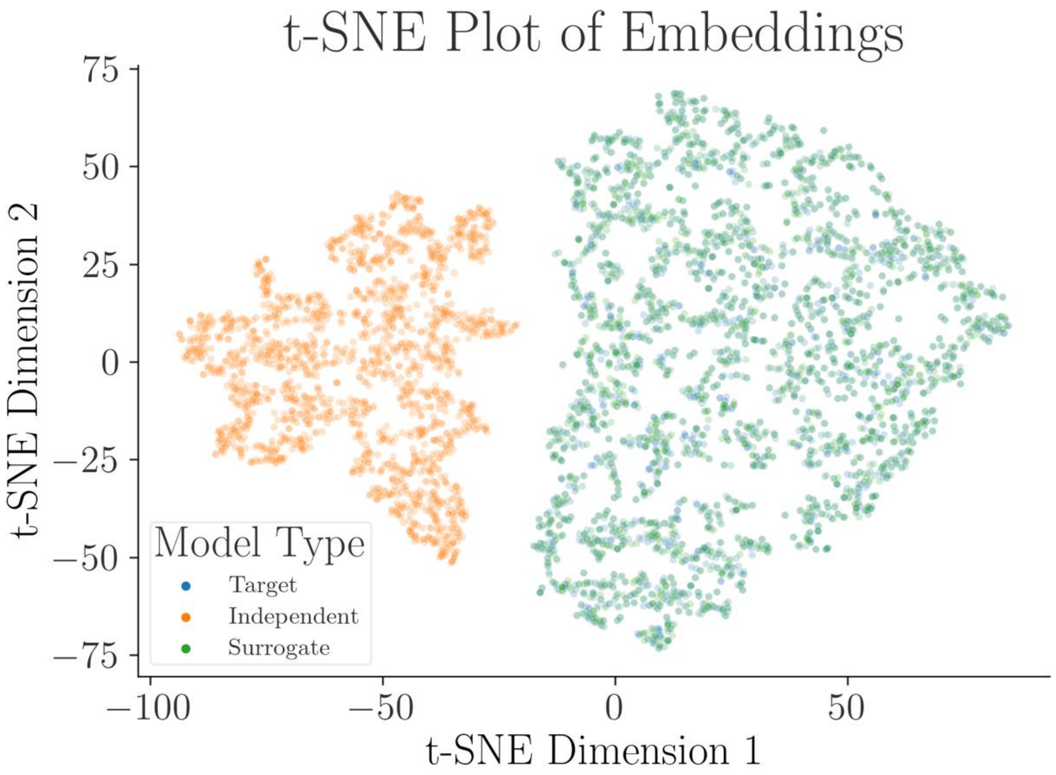
Does not degrade target model accuracy

Motivation

Unique embeddings for each input graph

High-fidelity model extraction
→ embeddings from surrogate and target models are similar

Can GNN embeddings be used as a fingerprint?



Threat Model

Blackbox Adversary (same as Shen et al.)

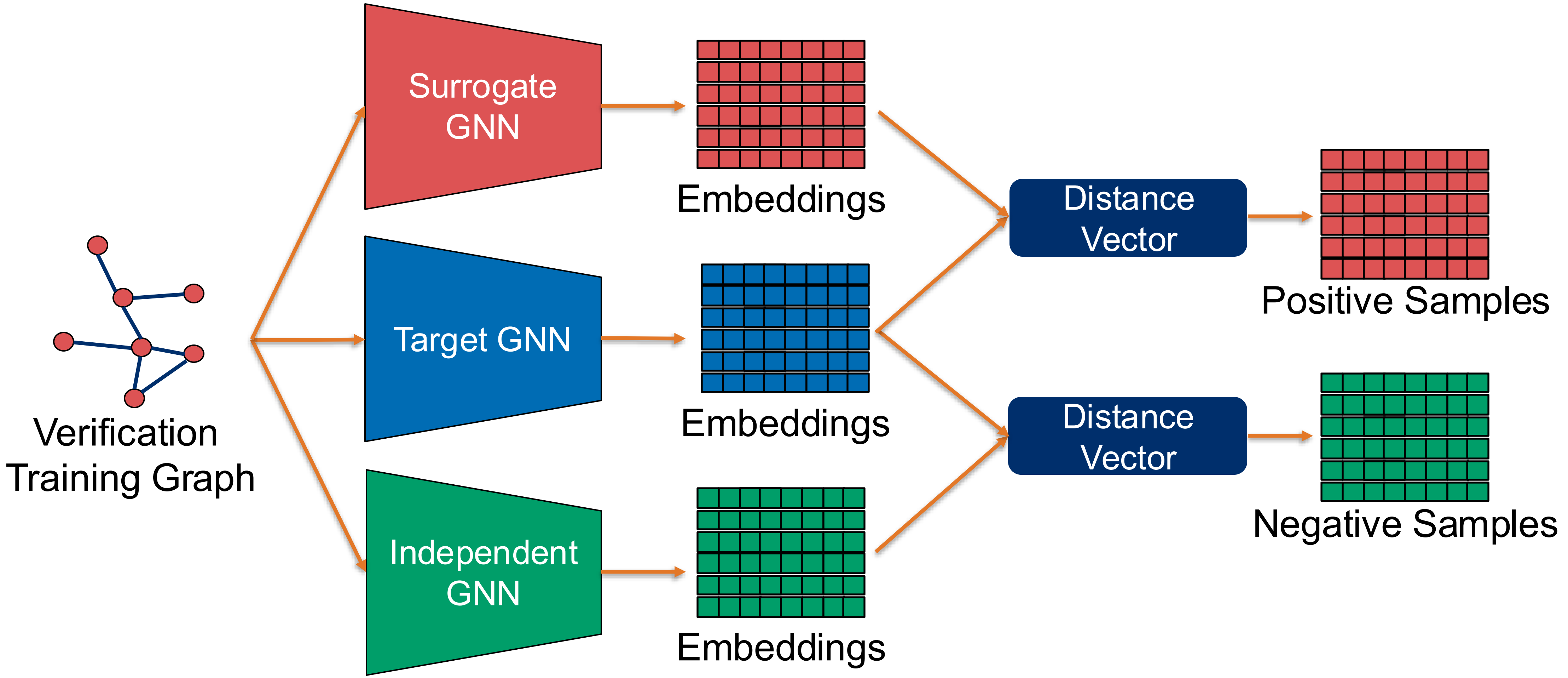
- Access to node embeddings to train surrogate model
- No overlap between surrogate and target training dataset

Ownership Verification

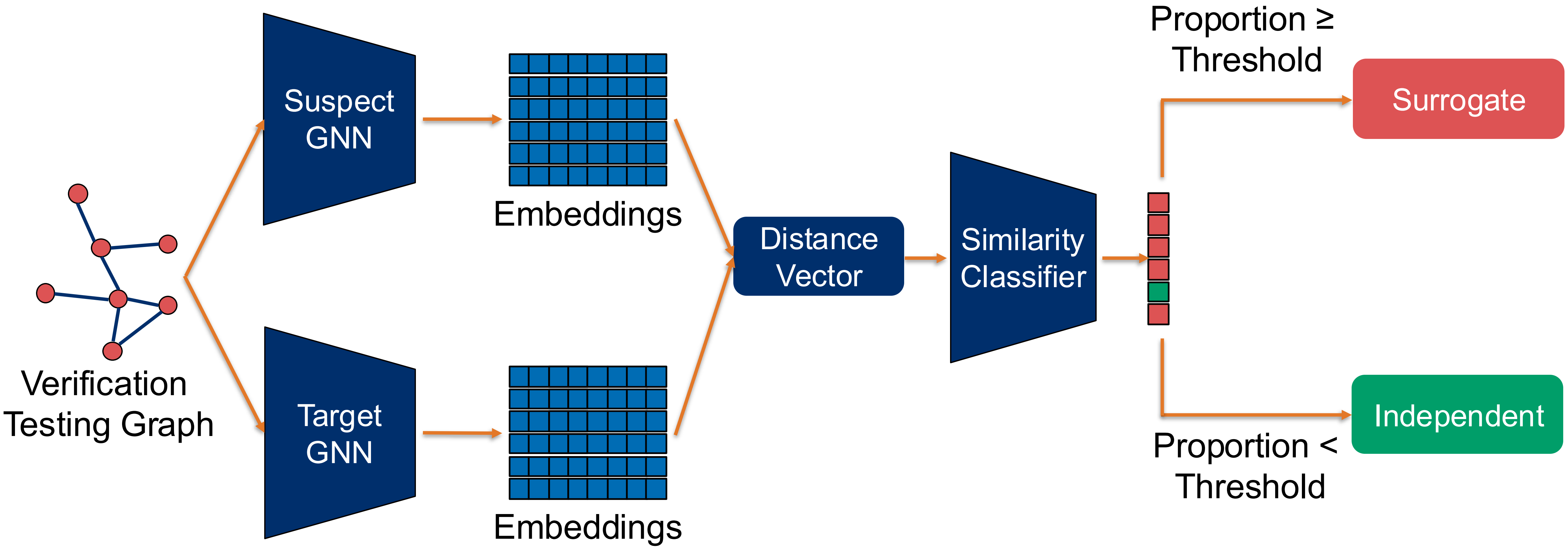
- Verifier samples verification dataset from same distribution as target model dataset
- Verifier can access target model and suspect model

Approach: Training Similarity Classifier (C_{sim})

Classify whether a pair of embeddings are close or far



Verification Steps



GroVE: Robustness

We consider only malicious suspects

Adversary can post-process surrogate models to evade detection

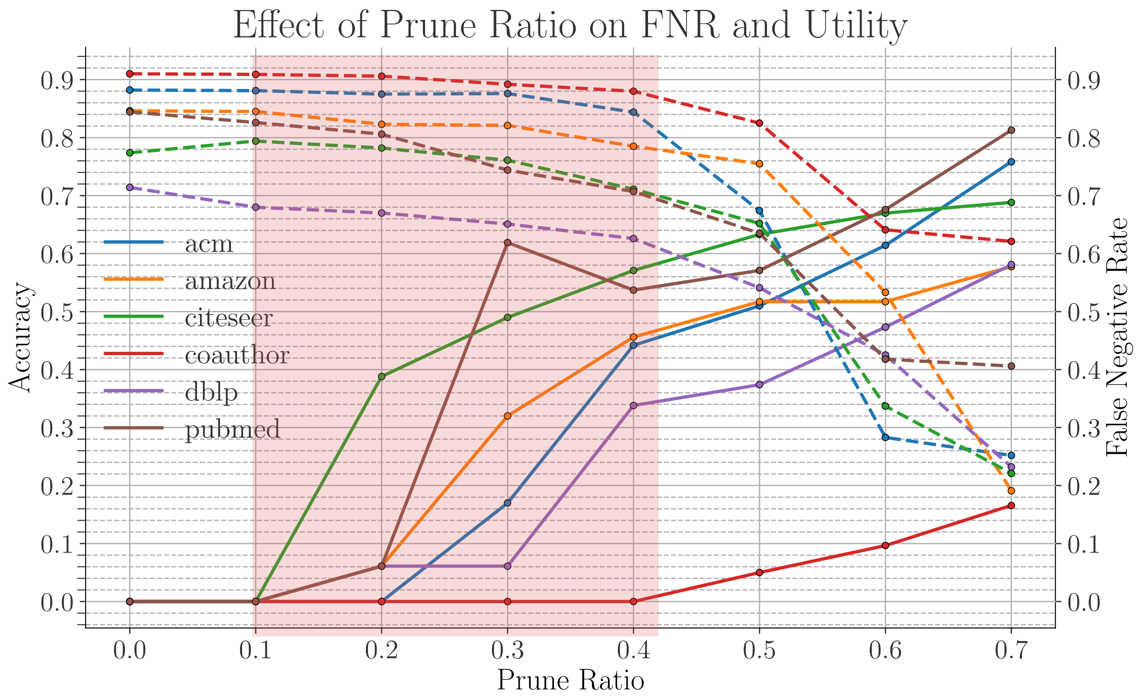
Possible post-processing techniques:

- Fine-tuning: GroVE is effective (zero FNR)
- Double Extraction: GroVE is effective (zero FNR)
- Pruning

Robustness: Pruning

Randomly remove some model weights
Changes the model’s embedding distribution

Pruning successfully evades GrOVe



Adversary wins: FNR increases without accuracy drop

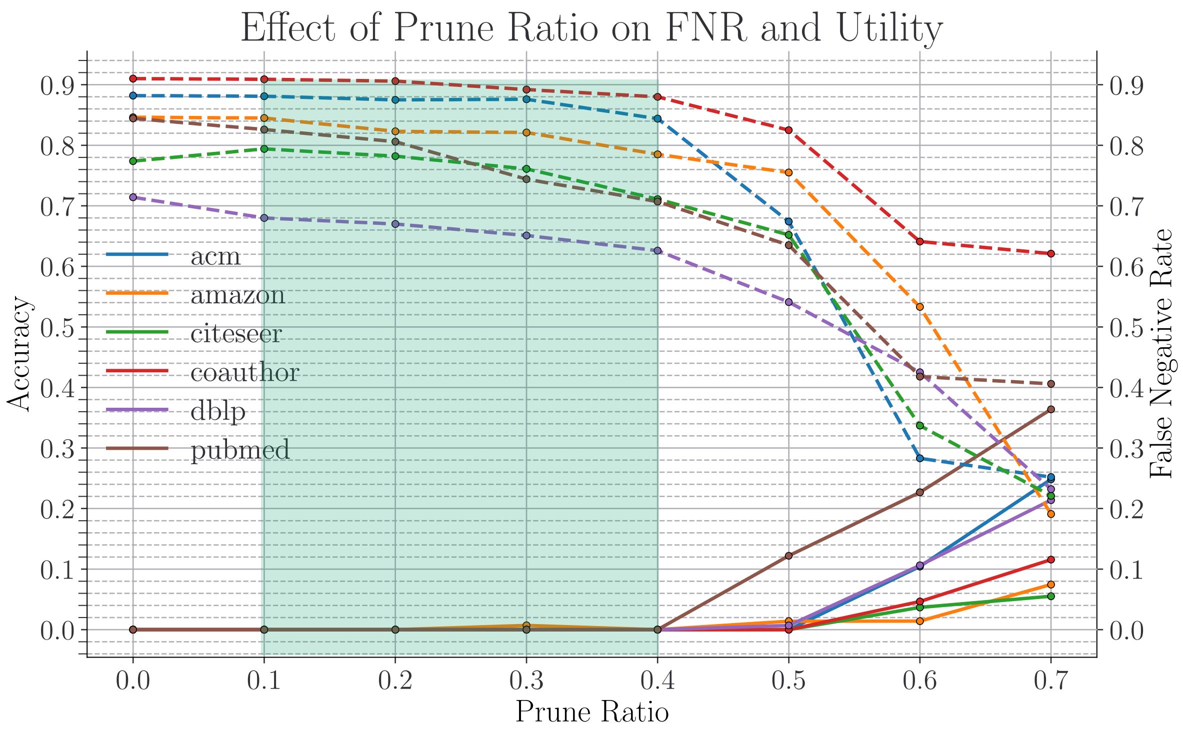
Making GrOVe Robust

Augment training data of C_{sim}

Include models with prune ratio <= 0.4 into training data

- 10% accuracy drop after 0.4

GrOVe after robust training correctly identifies surrogate models



No sweet spot for adversary

Takeaways

Model extraction attacks against GNNs are a problem

Surrogate models generate similar embeddings to target model

GrOVe is effective, robust, efficient, and accuracy

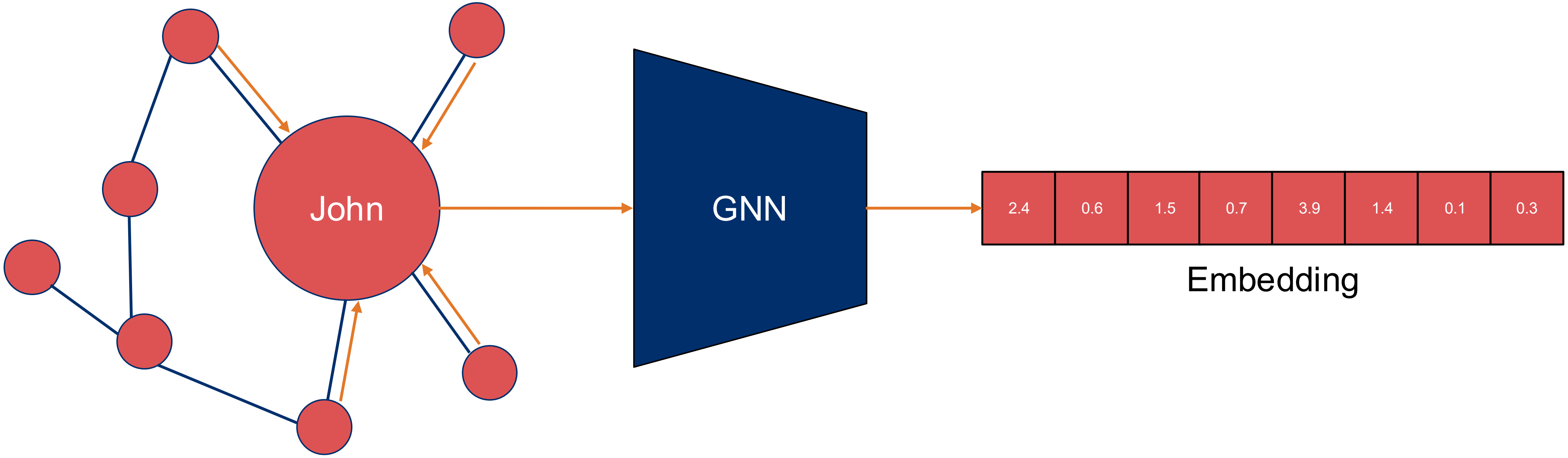
SCAN ME

https://arxiv.org/abs/2304.08566

Backup

Background: GNNs

Goal: Convert node features and graph structure to an embedding



Parties involved

Model owner

- Trains a model and deploys it as a service

Adversarial Responder ($\mathcal{Adv.R}$)

- Stole model from a model owner and wants to evade detection

Adversarial Accuser ($\mathcal{Adv.A}$)

- Wants to make false accusations against someone stealing their model

Third-party verifier ($\mathcal{Ver}$)

- Trusted third-party that verifies whether one model is stolen from the other

Model Registration

Goal: ensure $\mathcal{Ver}$ knows which model was trained first

Every model owner must:

- Generate cryptographic commitment (c) of their model
  - c should change if model changes (e.g., via cryptographic hash function)

- Obtain secure timestamp of c

Verification Process

Accuser claims that Responder stole their target model

$\mathcal{Ver}$:

1. checks that target and suspect models are consistent with registered models (including some additional checks)
2. checks the secured timestamps to ensure target model was trained before suspect model (preventing false accusations by $\mathcal{Adv.A}$)
3. samples verification dataset from same distribution as target model data
4. queries target and suspect model and passes outputs to verification algorithm

Embeddings as Fingerprints

Goal: Use embeddings to distinguish between surrogate and independent model

Steps:

- Train two models: target and independent
- Target model extraction with non-overlapping data to get surrogate model
- Query all three models with unseen verification graphs to generate embeddings

Model combinations:

- Training datasets: surrogate different, target and independent same
- Model architectures: different vs same architectures for all three models

Experiment 1

Goal: Analyze how embeddings are affected by model architecture and training data

Steps:

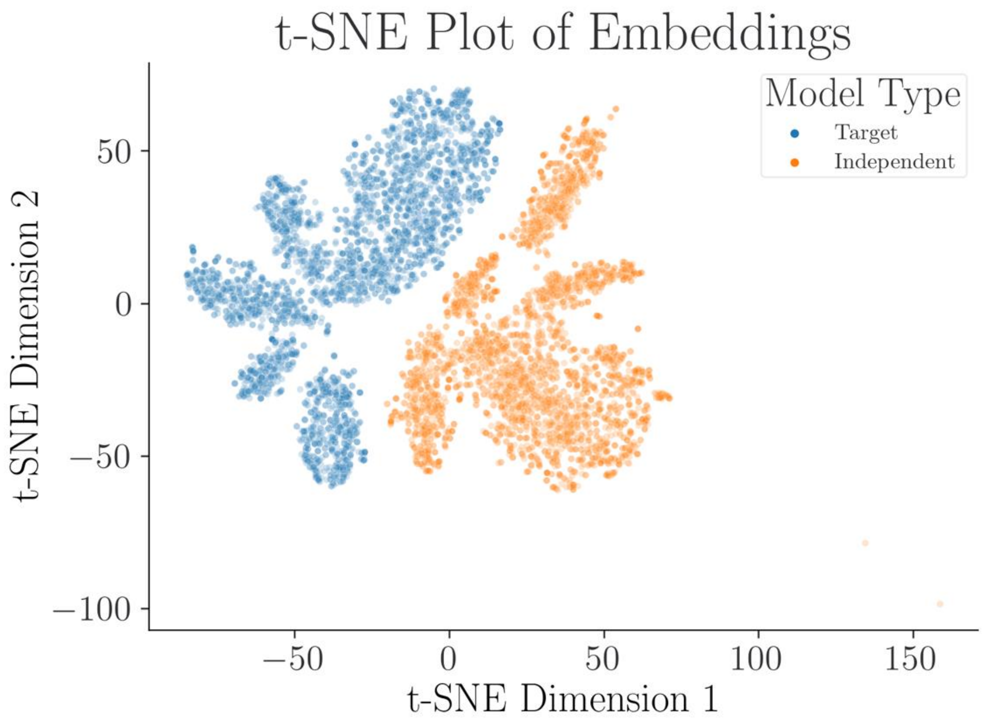
- Train two models: target and independent
- Query both with unseen verification graphs to generate embeddings
- Visualize 2D t-SNE projections of embeddings and compare distinguishability

Model combinations:

- Training datasets: different datasets of same distribution vs same dataset
  - 6 datasets: ACM, Amazon, Citeseer, Coauthor Physics, DBLP, and Pubmed
  - 10% data used for verification
- Model architectures: different vs same architectures
  - 3 architectures: Graph Attention Network (GAT), Graph Isomorphism Network (GIN), GraphSAGE (SAGE)

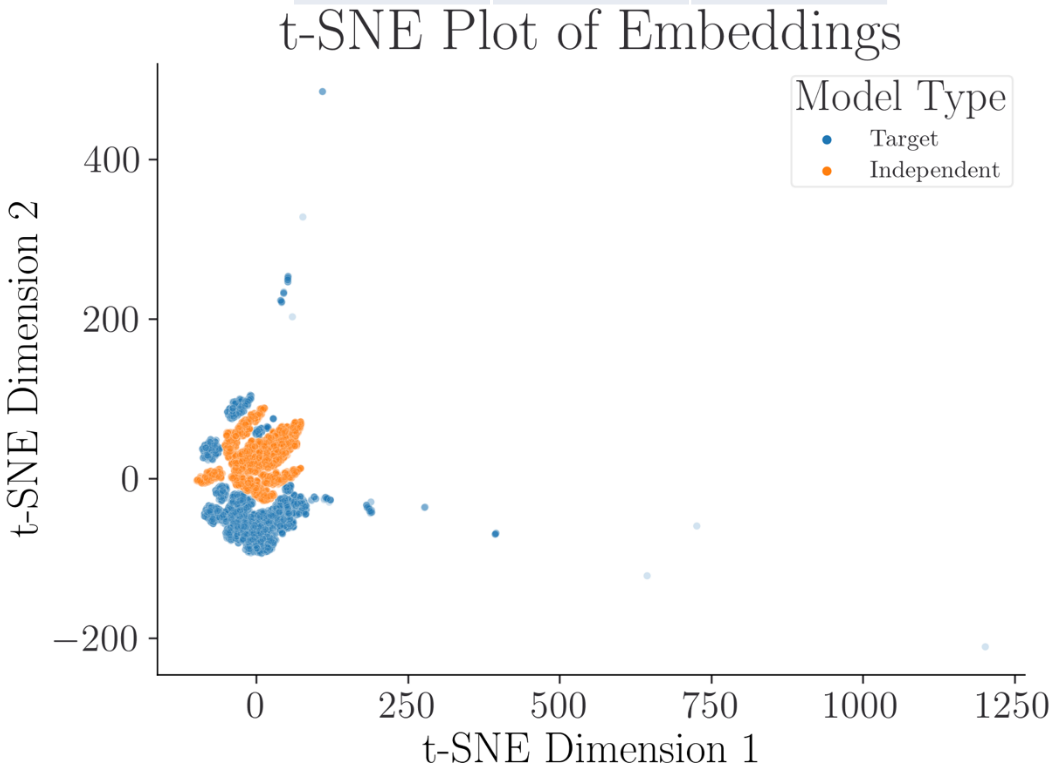
Experiment 1 Example Plots

	Architecture	Dataset
Target	GAT	coauthor1
Independent	GAT	coauthor2



Fully Separable

	Architecture	Dataset
Target	GIN	coauthor1
Independent	SAGE	coauthor2



Partially Separable

Experiment 1 Results

In all plots; **no overlap** between target and independent models

Different datasets:

- 54 total pairs, 4 are partially separable, rest are fully separable

Same dataset:

- 54 total pairs, 9 are partially separable, rest are fully separable

Experiment 1 Implications

Two models independently trained will always generate different embeddings

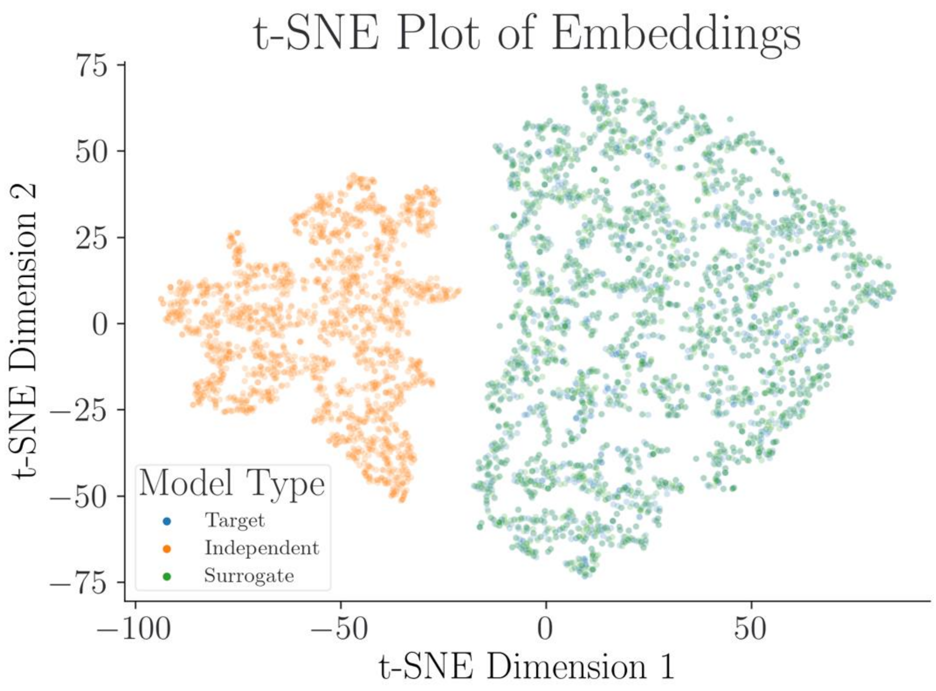
Same training data and same model architecture but different embeddings implies:

- Fingerprints based on embeddings cannot be used for dataset ownership verification

Can they be used for model ownership verification (detect a surrogate model)?

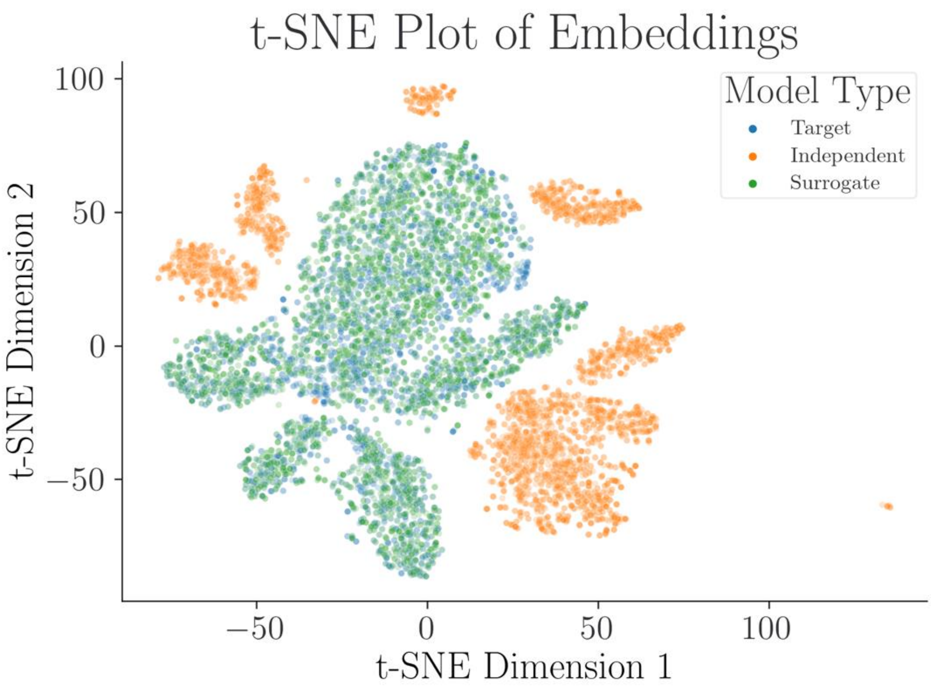
Visualizing Embeddings

	Architecture	Dataset
Target	GAT	pubmed1
Independent	GIN	pubmed1
Surrogate	GAT	pubmed2



Fully Separable

	Architecture	Dataset
Target	GAT	coauthor1
Independent	GIN	coauthor1
Surrogate	GAT	coauthor2



Partially Separable

Results

In all plots; target and surrogate model fully overlap

Independently trained model is in different space (fully separable)

Out of 30 models, in only 2 was independent model partially separable

Experimental Setup

Metrics

- Surrogate model accuracy
- False positive rate: Proportion of independent models misclassified as surrogate
- False negative rate: Proportion of surrogate models misclassified as independent

Training C_{sim}

- Type 1 model extraction attack for positive data points
- Independent models for negative data points

Testing C_{sim}

- Train additional independent and surrogate models using different random initializations

Model Extraction Results

Dataset	Target Accuracy	Independent Accuracy	Type 1 Surrogate Accuracy	Type 1 Surrogate Fidelity	Type 2 Surrogate Accuracy	Type 2 Surrogate Fidelity
acm	0.906 ± 0.025	0.919 ± 0.021	0.888 ± 0.019	0.931 ± 0.019	0.896 ± 0.010	0.954 ± 0.020
amazon	0.879 ± 0.064	0.876 ± 0.050	0.861 ± 0.022	0.870 ± 0.051	0.842 ± 0.007	0.848 ± 0.009
citeseer	0.804 ± 0.047	0.809 ± 0.028	0.757 ± 0.014	0.907 ± 0.041	0.796 ± 0.000	0.902 ± 0.012
coauthor	0.926 ± 0.005	0.928 ± 0.011	0.919 ± 0.019	0.949 ± 0.034	0.919 ± 0.004	0.948 ± 0.003
dblp	0.696 ± 0.028	0.693 ± 0.030	0.674 ± 0.009	0.833 ± 0.018	0.680 ± 0.008	0.851 ± 0.017
pubmed	0.846 ± 0.022	0.846 ± 0.021	0.829 ± 0.007	0.923 ± 0.016	0.832 ± 0.005	0.937 ± 0.014

Surrogate models consistent with attack paper

GroVE: Effectiveness

Dataset	FPR	Type 1 FNR	Type 2 FNR
acm	0.022 ± 0.022	0.000 ± 0.000	0.000 ± 0.000
amazon	0.034 ± 0.029	0.000 ± 0.000	0.000 ± 0.000
citeseer	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
coauthor	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
dblp	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
pubmed	0.002 ± 0.003	0.000 ± 0.000	0.000 ± 0.000

GrOVe is effective at verifying ownership

Robustness: Double Extraction

Adversary runs model extraction twice: against target model ➞ intermediate model; against intermediate model ➞ surrogate model

Intuition: Additional extraction changes the output distribution ➞ potentially evading GrOVe

Attack Type	Dataset	Surrogate Accuracy	Fidelity	FNR
Type 1	acm	0.843 ± 0.059	0.882 ± 0.060	0.000 ± 0.000
	amazon	0.776 ± 0.050	0.781 ± 0.063	0.000 ± 0.000
	citeseer	0.551 ± 0.140	0.627 ± 0.159	0.000 ± 0.000
	coauthor	0.924 ± 0.005	0.947 ± 0.012	0.000 ± 0.000
	dblp	0.686 ± 0.011	0.783 ± 0.011	0.000 ± 0.000
	pubmed	0.830 ± 0.007	0.912 ± 0.007	0.000 ± 0.000
Type 2	acm	0.882 ± 0.017	0.930 ± 0.020	0.000 ± 0.000
	amazon	0.698 ± 0.216	0.695 ± 0.219	0.000 ± 0.000
	citeseer	0.679 ± 0.064	0.736 ± 0.093	0.000 ± 0.000
	coauthor	0.916 ± 0.009	0.943 ± 0.004	0.000 ± 0.000
	dblp	0.678 ± 0.019	0.784 ± 0.036	0.000 ± 0.000
	pubmed	0.831 ± 0.004	0.930 ± 0.005	0.000 ± 0.000

GrOVe is effective at verifying ownership

GrOVe: Efficiency

Dataset	GAT		GIN		GraphSAGE	
	Generation	Train C_{sim}	Generation	Train C_{sim}	Generation	Train C_{sim}
acm	1184 ± 53	10562 ± 1548	1060 ± 55	10668 ± 1205	855 ± 34	10550 ± 1237
amazon	435 ± 25	3961 ± 492	418 ± 26	3845 ± 257	374 ± 25	3856 ± 288
citeseer	459 ± 30	4182 ± 462	412 ± 26	4011 ± 498	397 ± 25	3730 ± 202
coauthor	379 ± 26	3312 ± 218	361 ± 25	3273 ± 171	348 ± 21	3473 ± 323
dblp	389 ± 19	3312 ± 124	357 ± 24	3142 ± 204	349 ± 29	2970 ± 186
pubmed	334 ± 27	2985 ± 165	343 ± 27	2943 ± 223	351 ± 33	2876 ± 134

Total time to generate data and train C_{sim} < 3 hours

Influenced primarily by dataset size (Co-Author > DBLP > PubMed)