Adversarial Attacks on Neural Networks for Graph Data

Daniel Zügner, Amir Akbarnejad, Stephan Günnemann
Technical University of Munich, Germany

KDD 2018

Deep Learning on Graphs

Graphs are ubiquitous

- Social networks
- Web graphs
- Knowledge graphs

Deep learning on graphs

- (Unsupervised) learning of node representations (e.g. DeepWalk)
- Neural message passing (graph convolution)
- Implicit generative models for graphs

Attacks on Deep Learning for Graphs

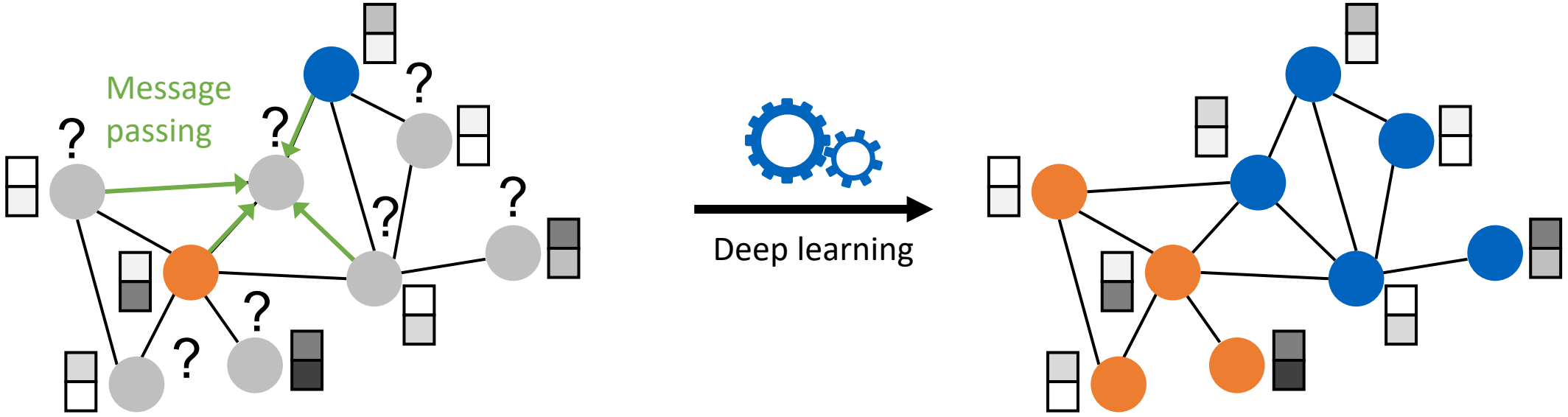
Adversaries are **very common** in application scenarios, e.g. search engines, or recommender systems.

These adversaries **will** exploit any vulnerabilities exposed.

In our work, we try to answer the question:

Are deep learning models for graphs robust with respect to adversarial attacks?

Semi-Supervised Node Classification



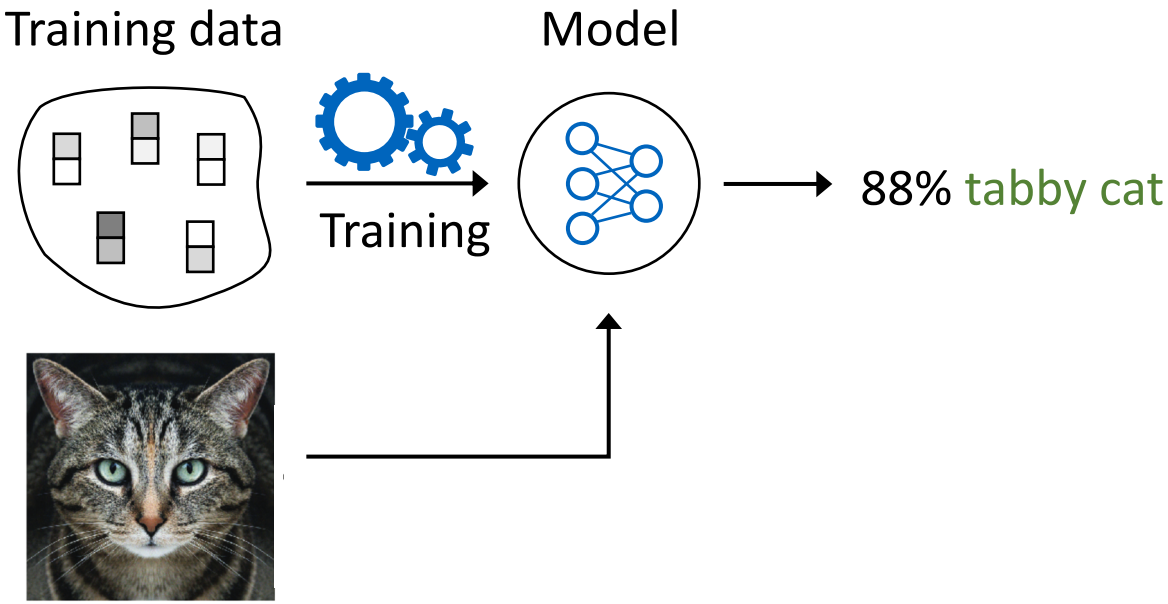
Partially labeled, attributed graph

Message passing (a.k.a. graph convolution) aggregates local information.

This could mean **higher robustness** – or lower robustness due to **cascading failures**!

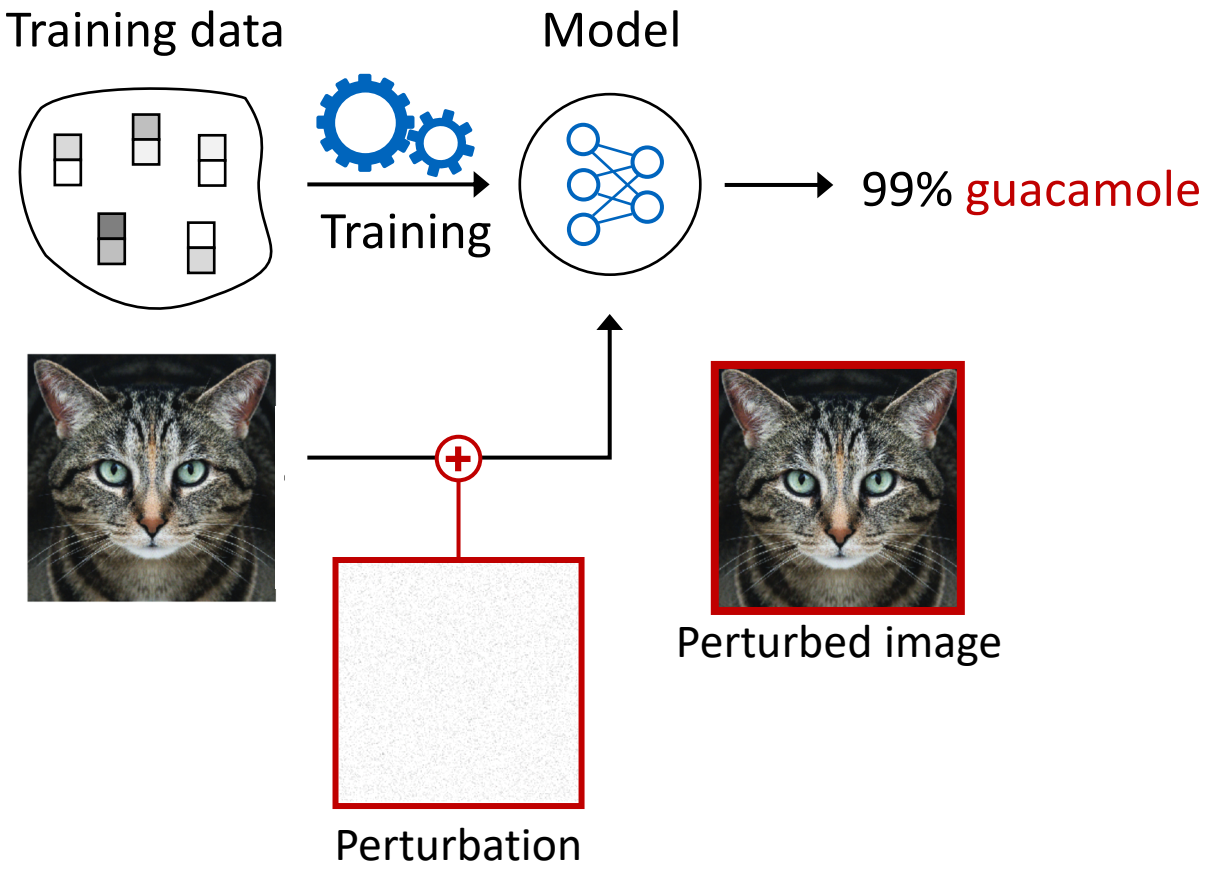
Real-world Example

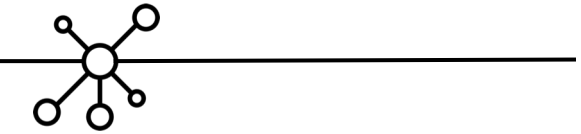
- Image of a tabby cat correctly classified



Real-world Example

- Image of a tabby cat correctly classified
- Add **imperceptible** perturbation
- Model classifies the cat as guacamole

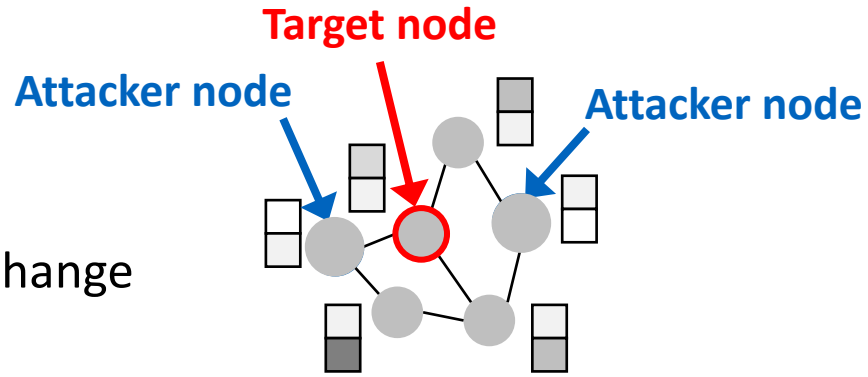




Adversarial attacks are a **real threat**.

When dealing with **graphs**, we need to **rethink possible attack** and defense **strategies**.

Attack possibilities

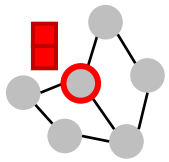


Target node $t \in V$: node whose classification label we want to change

Attacker nodes $S \subset V$: nodes the attacker can modify

Direct attack ($S = \{t\}$)

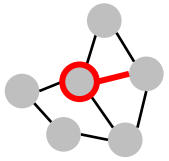
- Modify the **target**‘s features



Example

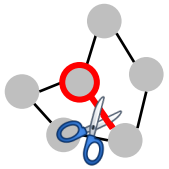
Change website content

- Add connections to the **target**



Buy likes/ followers

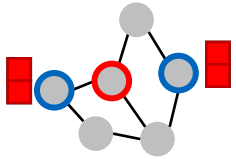
- Remove connections from the **target**



Unfollow untrusted users

Indirect attack ($t \notin S$)

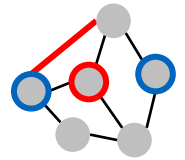
- Modify the **attackers**‘ features



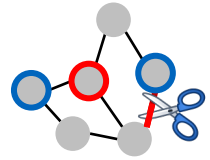
Example

Hijack friends of target

- Add connections to the **attackers**

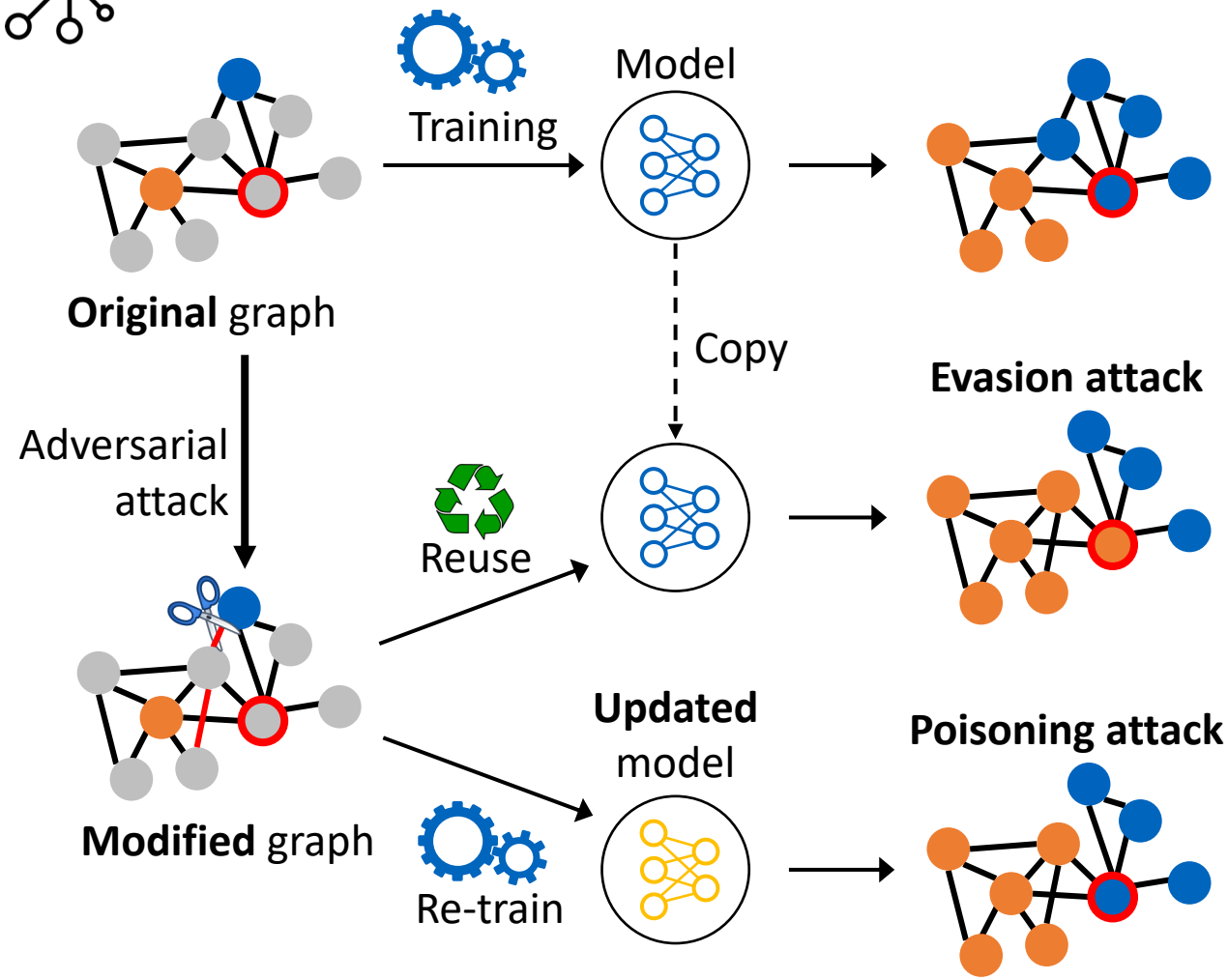


- Remove connections from the **attackers**



Create a link/ spam farm

Adversarial Attacks – Graphs

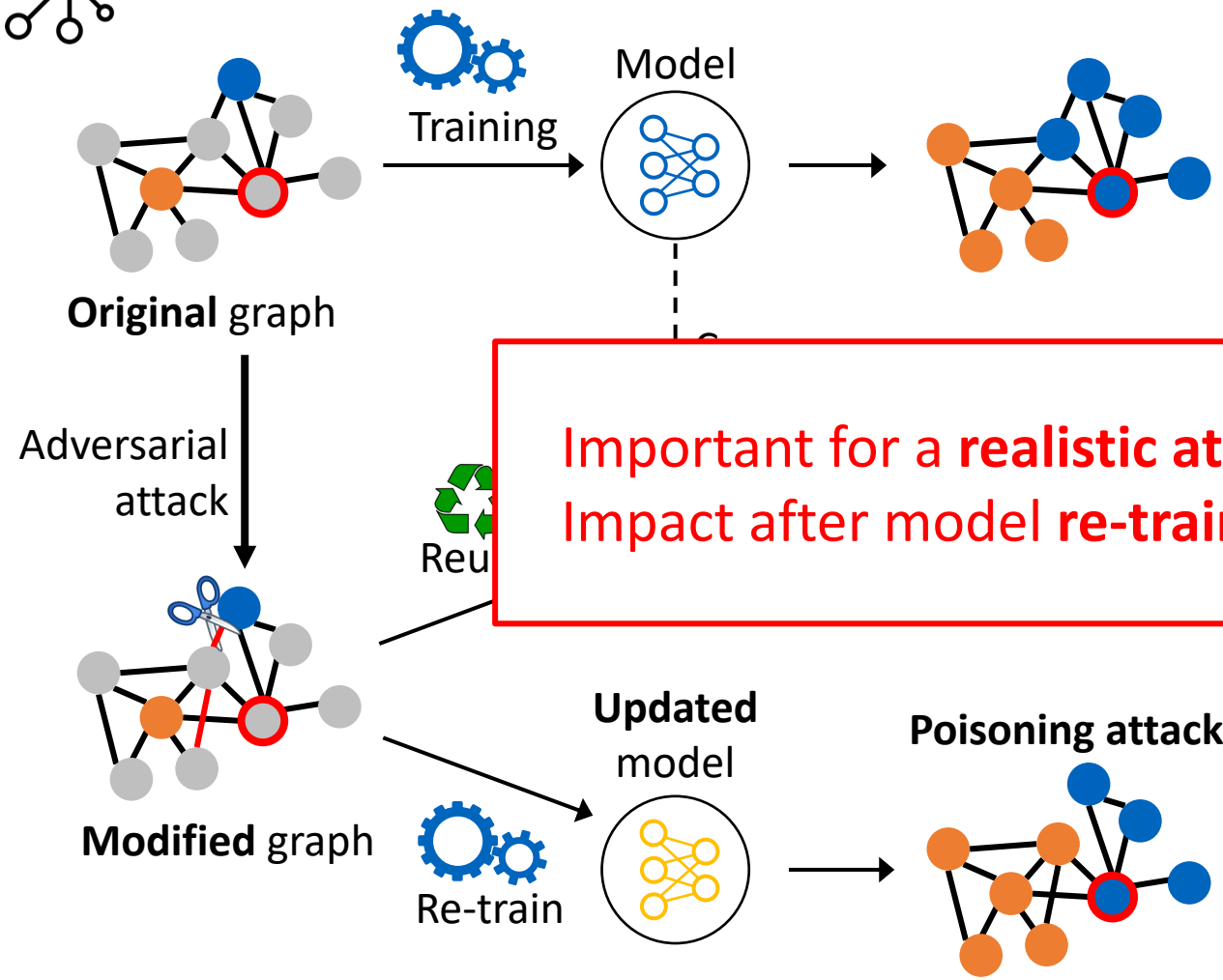


Transductive learning: data consists of labeled and unlabeled samples; all data used for training.

Evasion attack: Modify data to fool a static classifier.

But: modifications are on the **training data** (transductive setting).
Re-training can restore the predictions

Adversarial Attacks – Graphs



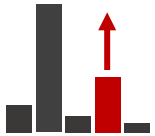
Transductive learning: data consists of labeled and unlabeled samples; all data used for training.

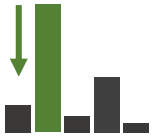
Important for a **realistic attack**:
Impact after model **re-training** (poisoning).

to fool a

But: modifications are on the **training data** (transductive setting).
Re-training can restore the predictions

Poisoning attack on node classification





$$\arg\max_{A',X'} \max_{c \neq c_{old}} \log Z^*_{v,c} - \log Z^*_{v,c_{old}}$$

$$where\ Z^* = f_{\theta^*}(A', X') = softmax\left(\hat{A}'\ ReLU\left(\hat{A}'X'W^{(1)}\right)W^{(2)}\right),$$

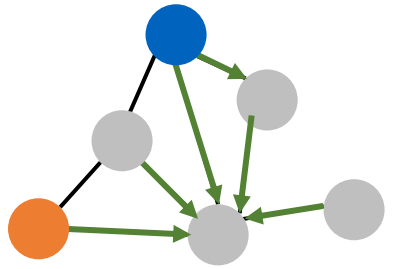
$$with\ \ \theta^* = \arg\min_{\theta} \mathcal{L}\,(\theta; A', X')$$

$$s.t.\,(A', X') \approx (A, X)$$

$A \in \{0,1\}^{N \times N}$: original adjacency matrix
$X \in \{0,1\}^{N \times D}$: (binary) node attributes
A': modified structure
X': modified features
v : target node

Poisoning attack on node classification

$$\arg\max_{A',X'} \max_{c \neq c_{old}} \log Z^*_{v,c} - \log Z^*_{v,c_{old}}$$



Message passing

$$where\ Z^* = f_{\theta^*}(A', X') = softmax\left(\hat{A}'\ ReLU\left(\hat{A}' X' W^{(1)}\right) W^{(2)}\right),$$

$$with\ \ \theta^* = \arg\min_{\theta} \mathcal{L}\,(\theta; A', X')\ \text{(after re-train)}$$

c.f. $\mathcal{L}(\theta; A, X)$: evasion

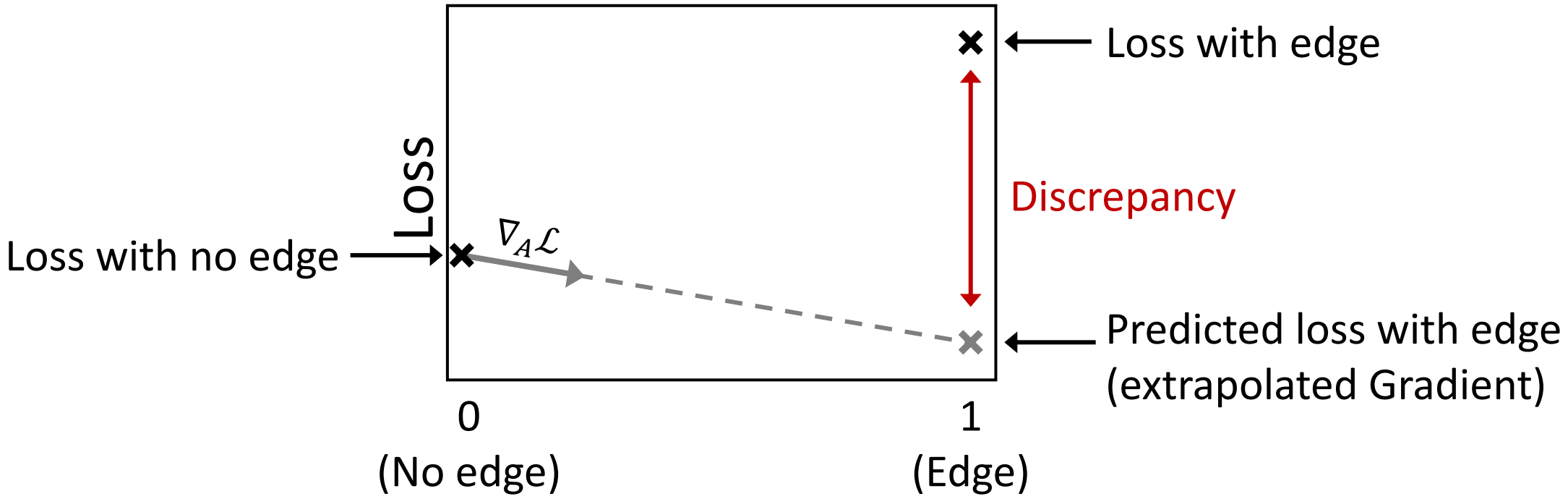
$A \in \{0,1\}^{N \times N}$: original adjacency matrix
$X \in \{0,1\}^{N \times D}$: (binary) node attributes
A': modified structure
X': modified features
v : target node

$$s.t.\ (A', X') \approx (A, X)$$

"Unnoticeability" constraint

Challenges in the Graph Setting

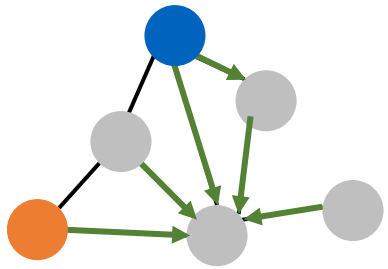
1. $\arg\max_{A',X'}$: optimization over **discrete variables** (gradient information less reliable)



Challenges in the Graph Setting

1. $\arg\max_{A',X'}$: optimization over **discrete variables**
   (gradient information less reliable)

2. Relational dependencies between the nodes: propagation effects



3. $(A', X') \approx (A, X)$:
   what is a sensible measure of '**closeness**' for (attributed) graphs?

4. $\theta^* = \arg\min_{\theta} \mathcal{L}(\theta; A', X')$:
   minimize classification accuracy **after** (re-)**training** on the modified data
   (transductive setting)

Our approach

- Use linear **surrogate model** to perform attacks efficiently while enforcing **unnoticeability** constraints on the changes.
- Train state-of-the-art **models** on the **perturbed data** and evaluate the **degradation in performance**.
- **No access** to the classifier is needed for the attack.

Surrogate Model

Based on a two-layer Graph Convolutional Network (GCN):

$$Z = f_\theta(A, X) = softmax\left(\hat{A}\, ReLU\left(\hat{A}XW^{(1)}\right)W^{(2)}\right)$$ **Linearize** classifier

$$\log Z' \quad = \hat{A}^2 XW'$$

Structure perturbations: $\max_{\hat{A}} \mathcal{L}'\left(\log Z'_v\right) where\ \log Z'_v = [\hat{A}^2 C]_v$ ← Constants

Feature perturbations: $\max_{X} \mathcal{L}'\left(\log Z'_v\right) where\ \log Z'_v = [C_1 X C_2]_v$

Surrogate Model

In contrast to the gradient, this simpler **surrogate model** allows us to **analytically** and **efficiently** determine the **exact impact** of a perturbation**.**

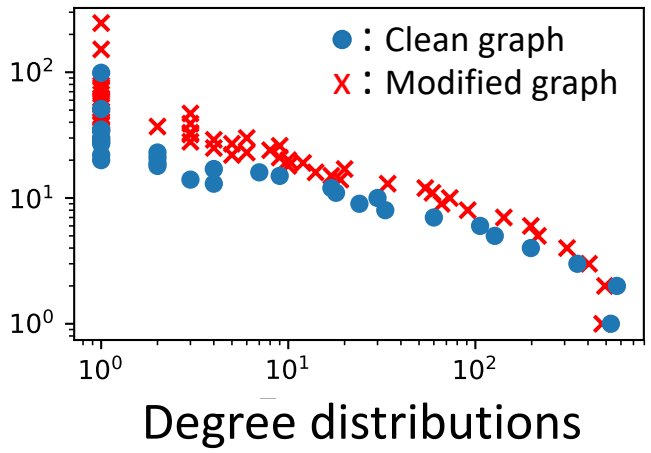
Unnoticeability Constraint

$(A', X') \approx (A, X)$: **Visual inspection** by a human **is not an option** for graphs.

What are sensible measures of '**closeness**' for graphs?

Structure perturbations: $A' \approx A$

Statistical test on the original and modified **degree distributions** to ensure structural similarity.



Degree distributions

Feature perturbations: $X' \approx X$

Co-occurrence constraint for features to prevent addition of unrealistic, easy to detect features

Adversarially inserted words to ML paper abstracts:

with constraint	**without constraint**
probabilistic	efforts
bayesian	david
inference	family

Unnoticeability Constraint

Closed-form equations**,** enabling a check for violations of the constraints in **constant time**.

Experiments

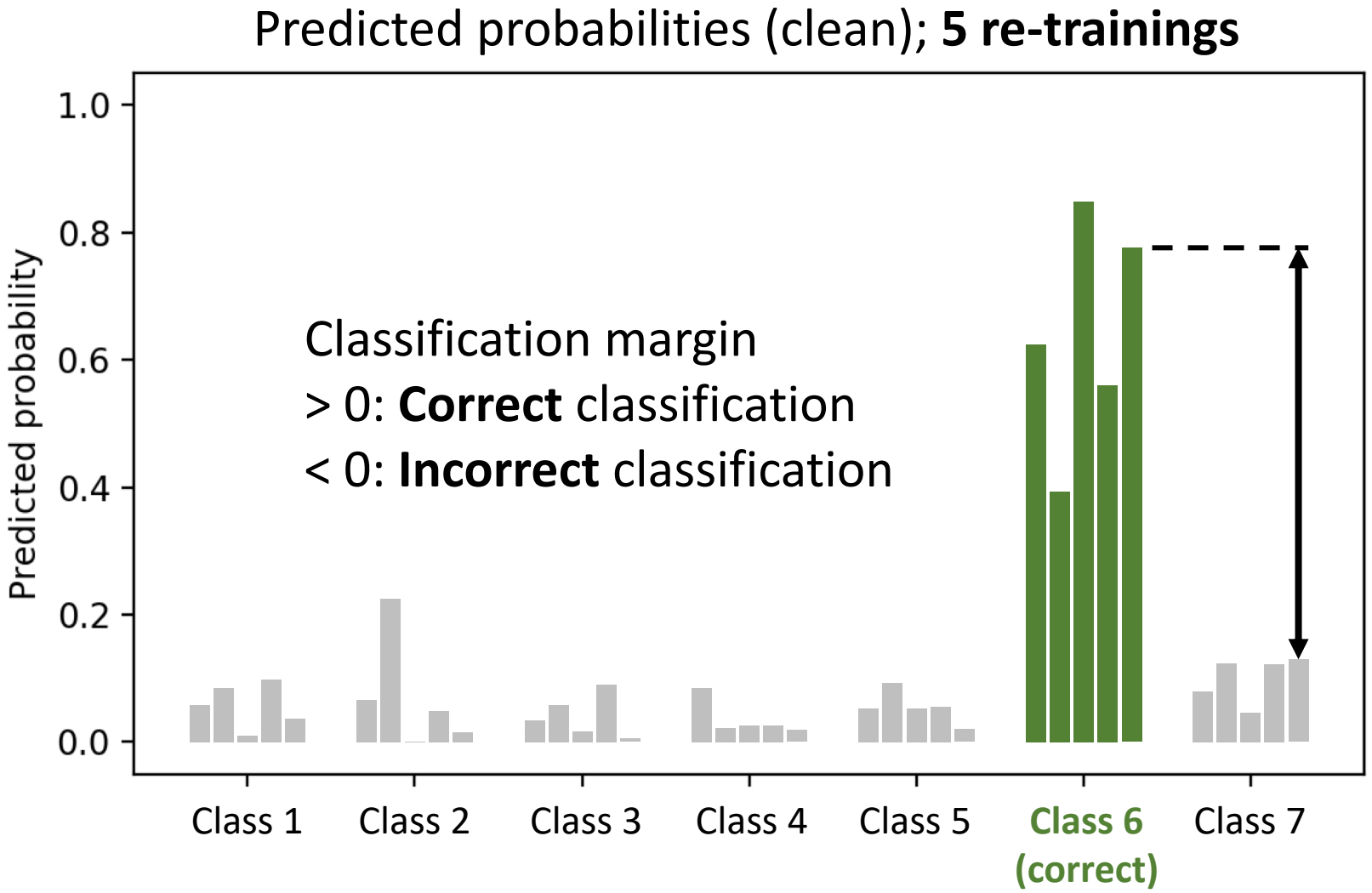
Dataset	Nodes	Edges	Attributes
Cora-ML	2,485	5,069	1,433
Citeseer	2110	3,668	3,703
PolBlogs	1,222	16,714	–

- Experimental evaluation on three datasets: Cora (ML), Citeseer, PolBlogs
- Transfer experiments with **Graph Convolutional Network** (GCN), **Column Network**, and **DeepWalk**
- **Perturbation budget is d+2**, where d is the target‘s degree
- Evaluation on **5 different splits**; **10x re-training** per attack

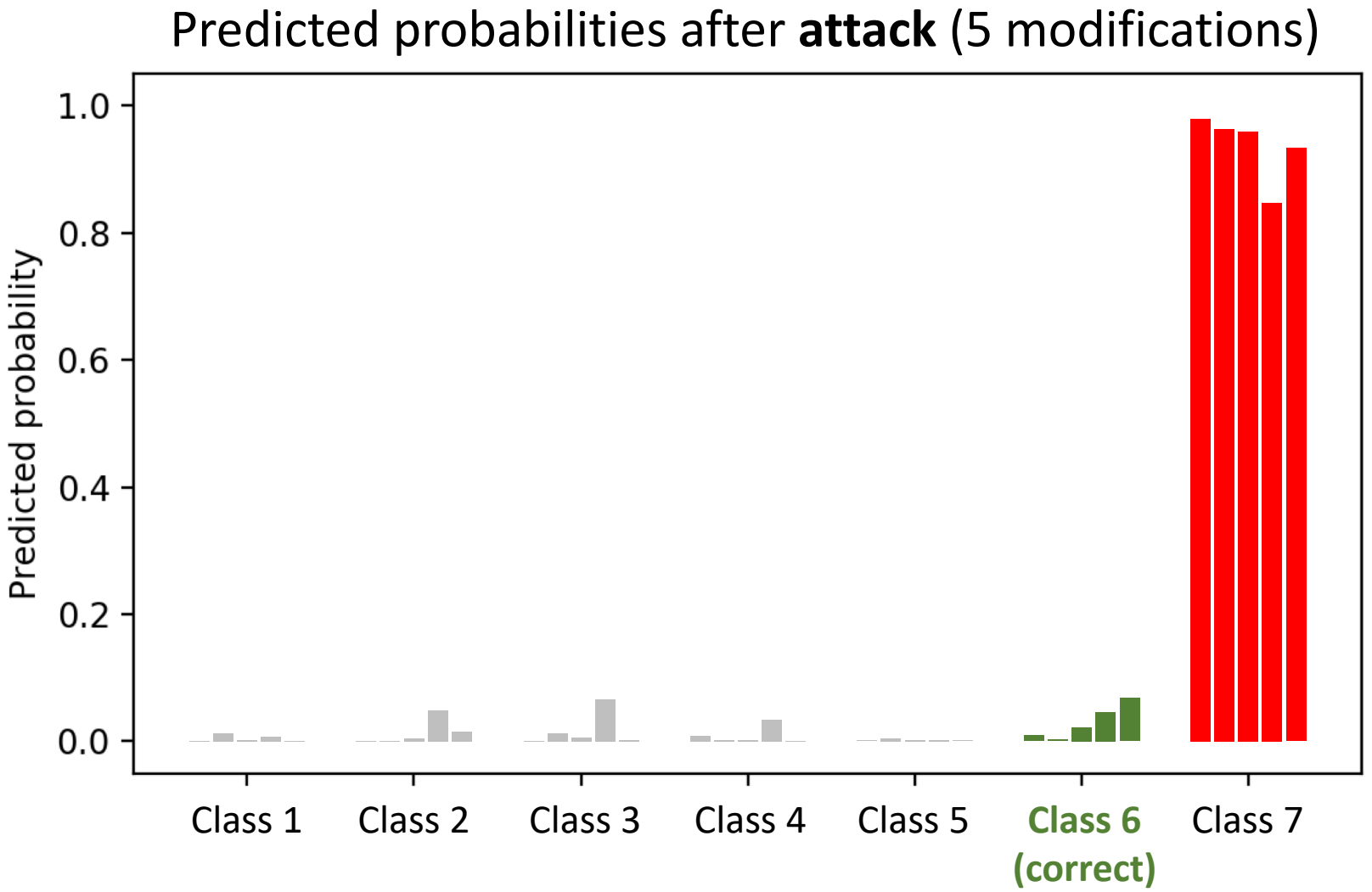
Baselines:

- **Inter-class random** (direct; structure): insert edges randomly to nodes from different classes.
- **Gradient** (direct; structure): insert/remove edges based on the gradient.

Results: Example Attack on GCN

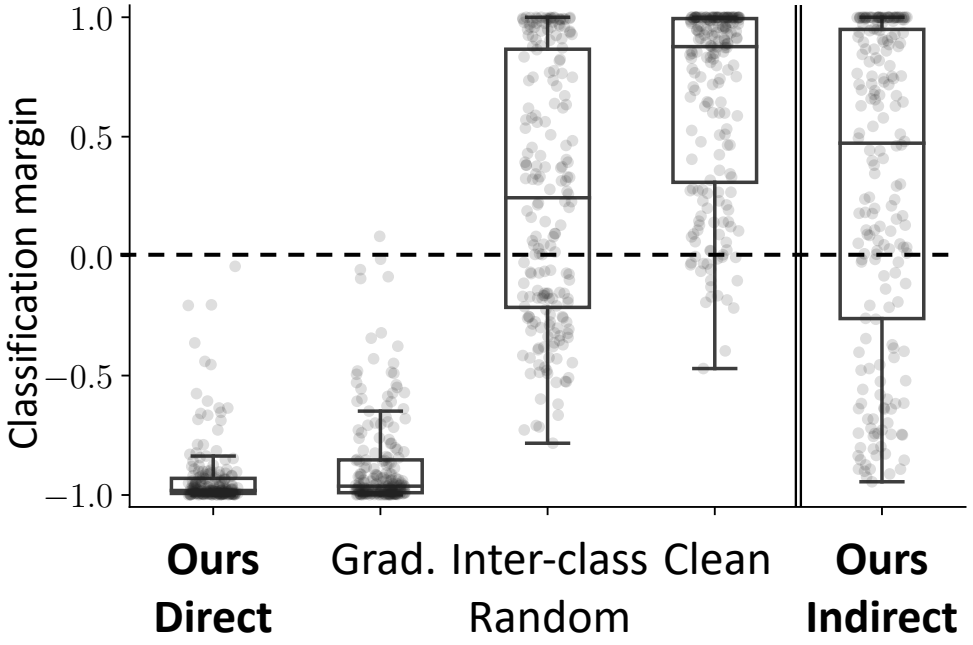


Results: Example Attack on GCN

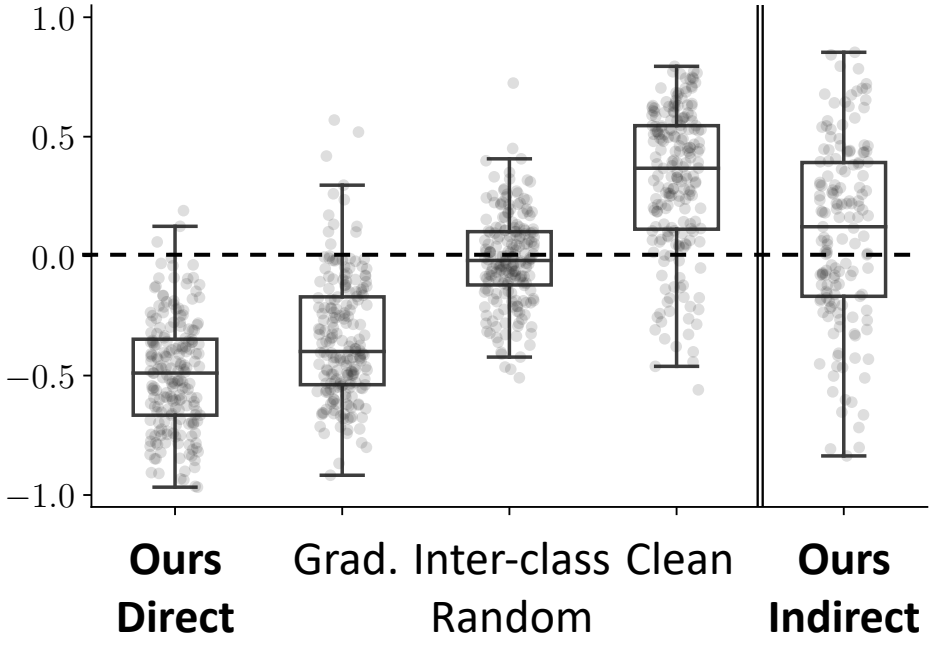


Results: Transfer

Poisoning attack on **GCN**



Poisoning attack on **DeepWalk**



Correct classification

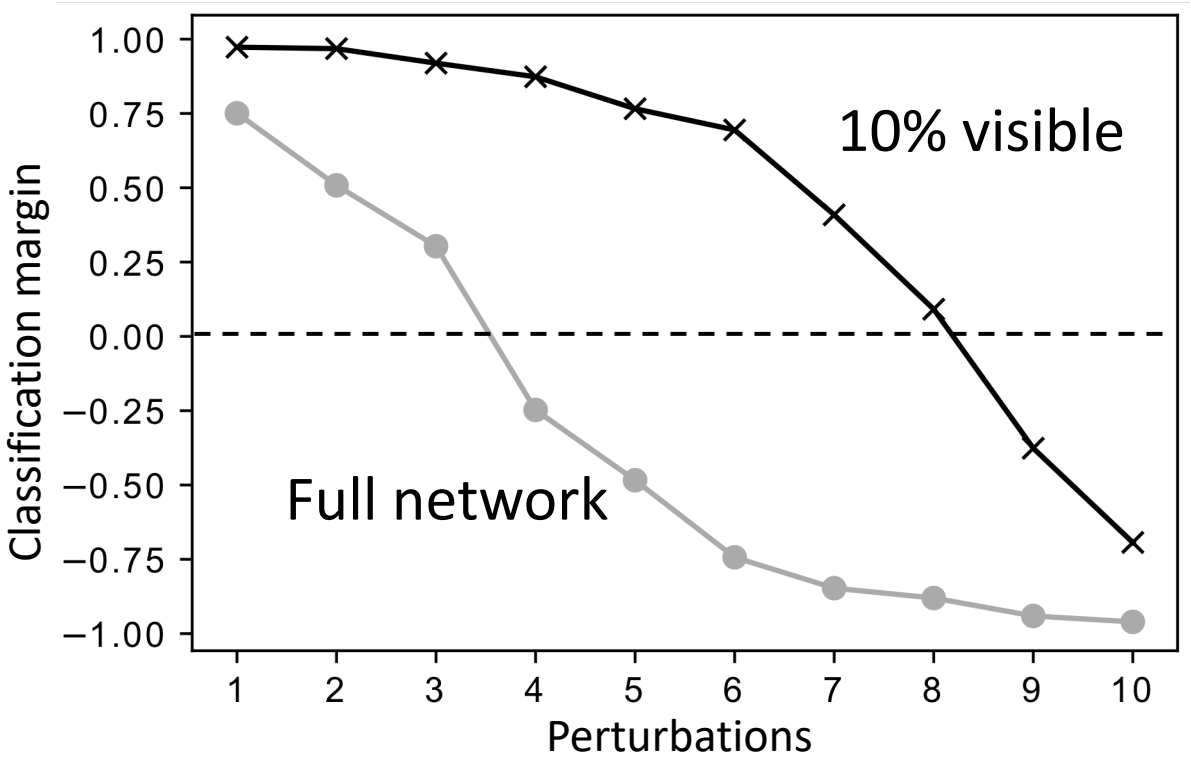
Wrong classification

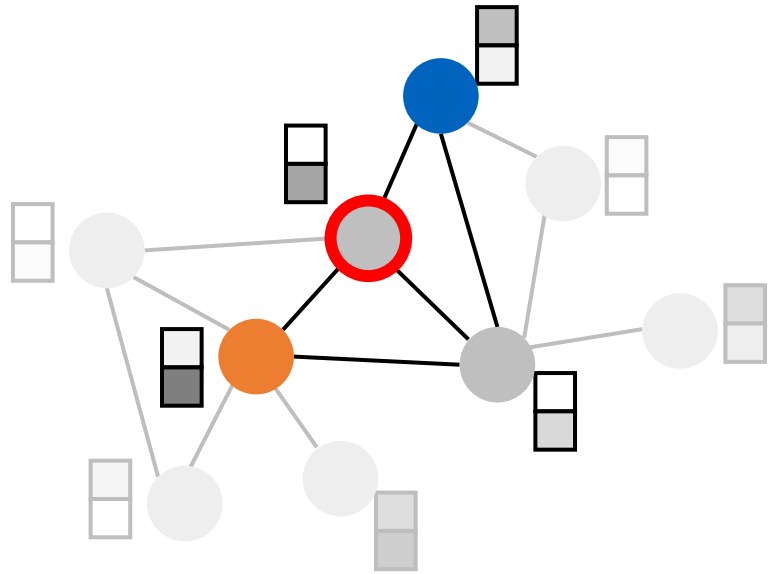
	Ours Direct	Grad.	Inter-class Random	Clean	**Ours Indirect**
% Correct:	**1.0%**	2.7%	60.8%	90.3%	67.2%
	Ours Direct	Grad.	Inter-class Random	Clean	**Ours Indirect**
	2.1%	9.8%	46.2%	83.8%	59.2%

Deep learning models for graphs are **not robust** to adversarial attacks.

Results: Limited knowledge

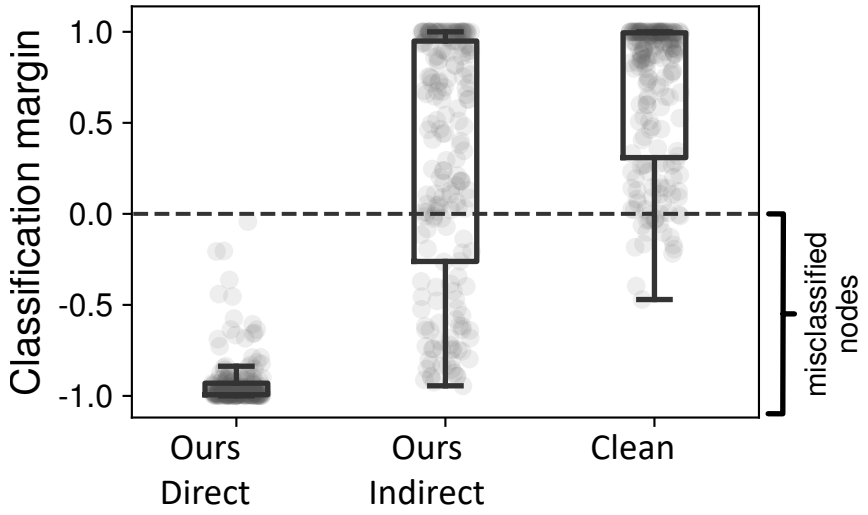
Setup: Only provide a **small part of the network** around the target node to the surrogate model to attack (evaluation with GCN on the complete graph).





Conclusion

- Deep learning models for graphs are **highly vulnerable** to adversarial attacks.
- We propose an **efficient algorithm** for performing **transferable attacks**.
- These attacks are **successful** even under **restrictive attack scenarios**, e.g. no access to target node or limited knowledge about the graph.



github.com/danielzuegner/nettack