Unveiling the Threat of Fraud Gangs to Graph Neural Networks: Multi-Target Graph Injection Attacks Against GNN-Based Fraud Detectors

Jinhyeok Choi, Heehyeon Kim, and Joyce Jiyoung Whang*
School of Computing, KAIST
* Corresponding Author

The 39th AAAI Conference on Artificial Intelligence
(AAAI 2025)

KAIST BDILab
BIG DATA INTELLIGENCE

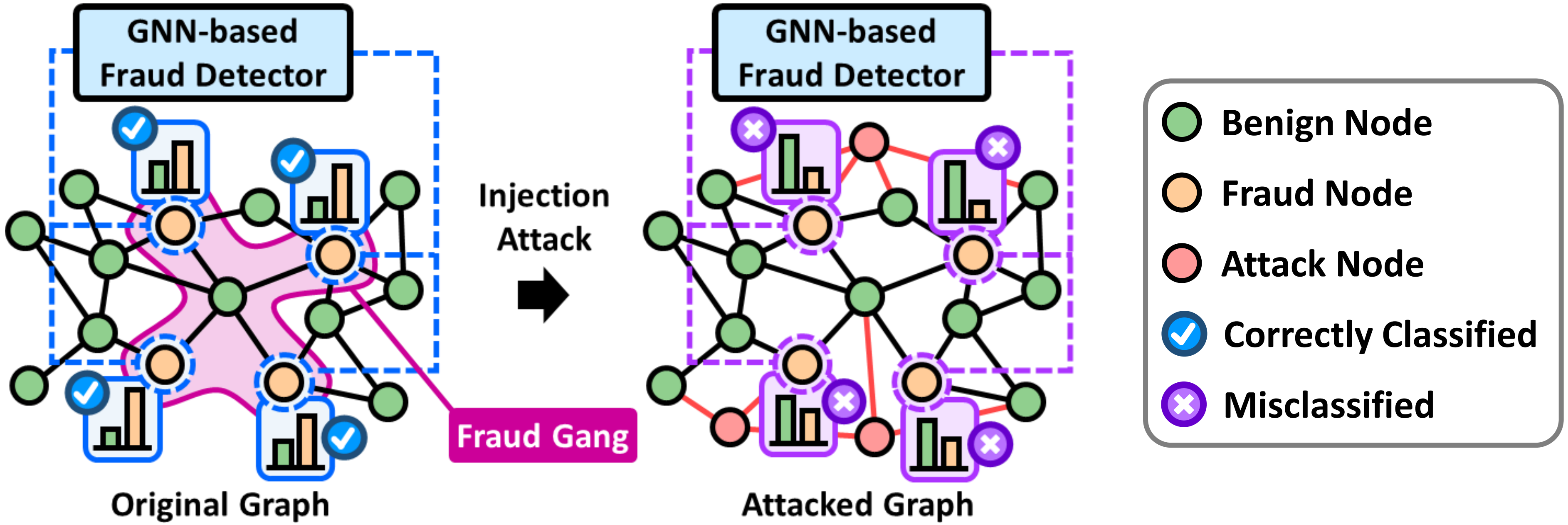
01 Fraud Detection with GNNs

- **Complex interactions of fraudsters** can be effectively modeled using **graphs**
  - **Frauds** are typically represented as **nodes** corresponding to individuals with malicious intentions.
- **Various tailored GNNs** have recently been introduced to filter **the camouflaged fraudsters**
  - e.g., CARE-GNN (CIKM 2020), PC-GNN (TheWebConf 2021), GAGA (TheWebConf 2023)
- **Vulnerabilities** of the GNN-based fraud detectors **to adversarial attacks** remain **unexplored**

- Frauds are increasingly **organized into gangs or groups**
  - **Fake news**: fraudsters can spread misinformation by using multiple fake accounts
  - **Spam reviews**: fraudsters could create multiple fake reviews using different IDs
  - **Medical insurance frauds**: fraudsters may collaborate with doctors or insurance agents to obtain fake diagnoses
- Define the adversarial attack on GNN-based fraud detectors as a **multi-target graph injection attack**
  - **Nodes** represent **distinct entities** such as news, reviews, and claims, and **edges** represent **their relationships**

01 Attack Scenarios

- Adopt a **graph injection attack**, as it is **more feasible than a graph modification attack**, which requires privileged access to alter existing structures
- Consider a **black-box evasion** attack to conduct attacks in the most realistic setting
  - **Black-box attack**: the attacker **can access only the original graph, partial labels, and a surrogate model**
  - **Evasion attack**: the attack occurs during the victim model's **inference phase**



01 Limitations of Existing Methods

- Inject multiple attack nodes **sequentially**, **fixing the graph structure at each step**
  - **Limits** their **flexibility and efficiency** in exploring diverse structures, as **it requires to fix the degree budget across all attack nodes due to a lack of information about future steps**

- Focus on **single-target or single injection attacks**, and often **overlook interactions within target nodes and among attack nodes**

- **Sequentially generate attributes and edges** of attack nodes, considering **only one-way dependency**

01 Contributions

- Investigate **adversarial attacks against GNN-based fraud detectors** by fraud gangs
  - First study on graph injection attacks **against GNN-based fraud detectors**
  - First study on graph injection attacks **for multiple target nodes organized by groups**
  - Create datasets and target sets grouped based on metadata or relations in real-world graphs

- Propose **MonTi**, a transformer-based **M**ulti-target **one**-**T**ime graph **i**njection attack model
  - Flexible and efficient attacks by **adaptively allocating degree budgets** and injecting all attack nodes **at once**
  - Capture **interdependencies between node attributes and edges**
  - Consider **interactions within target nodes** and **among attack nodes**

- MonTi substantially outperforms state-of-the-art graph injection attack methods in **both multi- and single-target settings** on real-world graphs

02 GNN-Based Fraud Detectors

An undirected attributed graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{X})$

- $\mathcal{V}$ is a set of n nodes, $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is a set of edges, $\mathcal{X}$ is a set of node attribute vectors
  - $\mathbf{x}_v \in \mathbb{R}^D$ represents **the attribute vector of a node** $v \in \mathcal{V}$
- $\mathcal{Y}$ denotes a set of node labels
  - $y_v \in \{0,1\}$ represents **the label of a node** $v \in \mathcal{V}$ where $y_v = 1$ indicates v is a fraud node

A GNN-based fraud detector $f_\theta(\cdot)$

- θ indicates learnable parameters
- The **fraud score vector** $\mathbf{s}_v = f_\theta(G, v) = \mathrm{MLP}\big(\Phi(G, v)\big) \in \mathbb{R}^2$
  - $\Phi(\cdot)$ denote a GNN encoder
- The **predicted label** $\hat{y}_v = \arg\max_i s_{v,i} \in \{0,1\}$
  - $s_{v,i}$ denotes the i-th element of $\mathbf{s}_v$

02 Multi-Target Graph Injection Attacks

A graph injection attack injects attack nodes $\mathcal{V}_{\text{in}}$ with attributes $\mathcal{X}_{\text{in}}$ and edges $\mathcal{E}_{\text{in}}$ into a graph G

- The perturbed graph $G' = (\mathcal{V}', \mathcal{E}', \mathcal{X}')$
  - $\mathcal{V}' = \mathcal{V} \cup \mathcal{V}_{\text{in}}, \mathcal{E}' = \mathcal{E} \cup \mathcal{E}_{\text{in}}, \mathcal{X}' = \mathcal{X} \cup \mathcal{X}_{\text{in}}$ where $\mathcal{E}_{\text{in}} \subset (\mathcal{V}' \times \mathcal{V}') \setminus (\mathcal{V} \times \mathcal{V})$
  - Does not modify existing nodes and edges

The multi-target graph injection attack against a GNN-based fraud detector

- **Target set $\mathcal{T} \subset \mathcal{V}$** consisting of **fraud nodes**
- The objective function

Prediction of the trained fraud detector on a target node after attack

Node budget

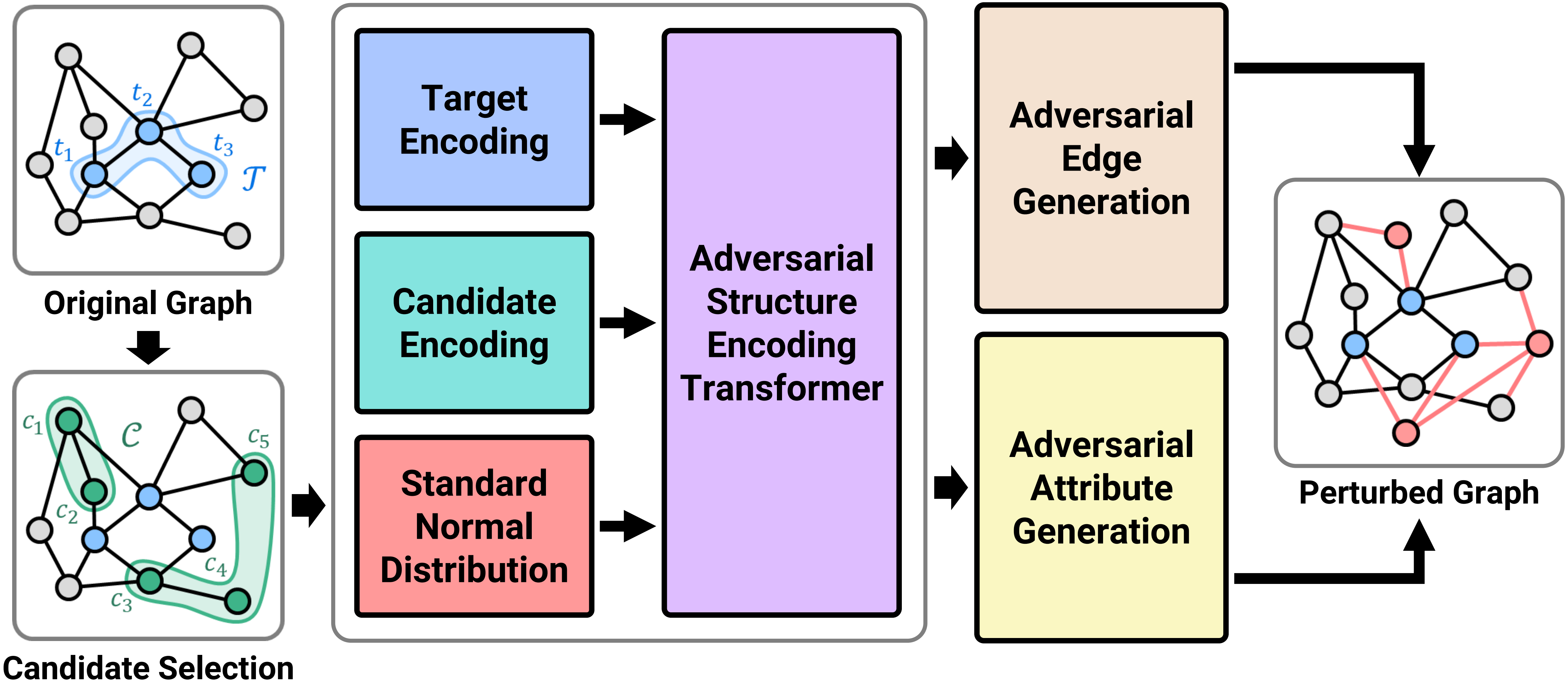
Edge budget

$$\min_{G'} \sum_{t \in \mathcal{T}} \mathbb{I}\left(\boxed{\arg\max_i s'_{t,i}} = y_t\right) \ \text{s.t.}\ |\mathcal{V}_{\text{in}}| \leq \Delta, |\mathcal{E}_{\text{in}}| \leq \eta$$

  - $\mathbf{s}'_v = f_{\theta^*}(G', v)$ where $\theta^* = \text{argmin}_\theta\, \mathcal{L}_{\text{train}}(f_\theta, G, \mathcal{D})$, $\mathcal{L}_{\text{train}}$ is a training loss of $f_\theta(\cdot)$, and $\mathcal{D} \subset \mathcal{Y}$ is a training label set

03 MonTi: Multi-Target One-Time Graph Injection Attack Model

Overview of MonTi

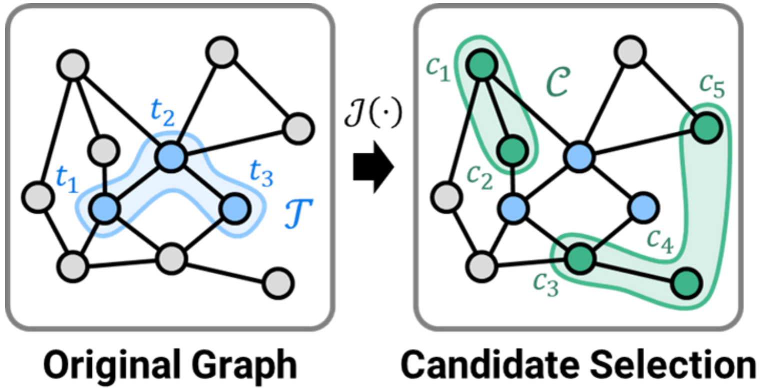


03 Candidate Selection

Three types of contexts that affect multi-target graph injection attacks

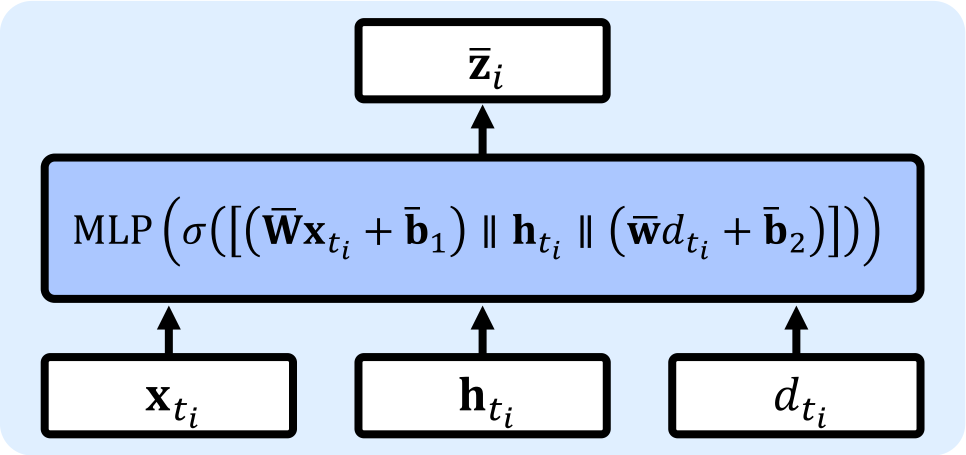
- **Target nodes** $\mathcal{T} = \{t_1, \cdots, t_m\}$, **Candidate nodes** $\mathcal{C} = \{c_1, \cdots, c_\alpha\} \subset \mathcal{N}^{(K)}$, and **Attack nodes** $\mathcal{V}_{\text{in}} = \{u_1, \cdots, u_\Delta\}$
  - $\mathcal{N}^{(K)}$ is a set of K-hop neighbors of the target nodes, excluding the target nodes themselves

A learnable candidate scoring function $\mathcal{J}(\cdot)$

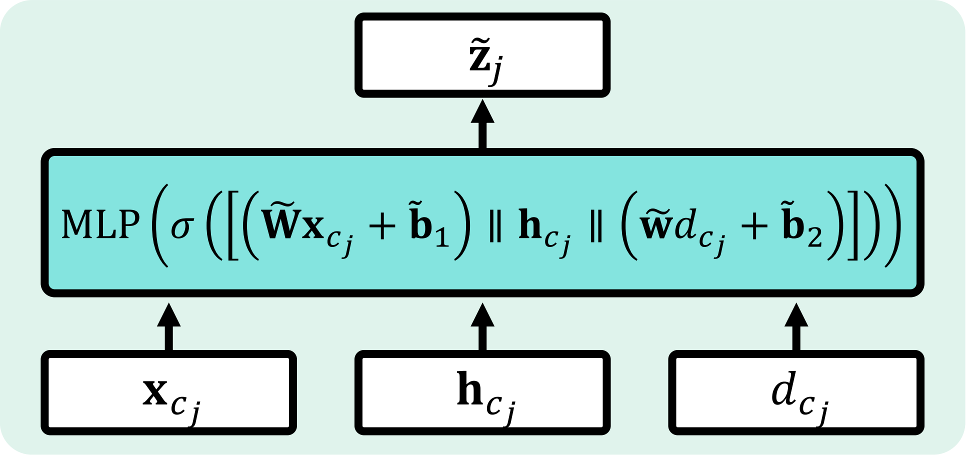


- $|\mathcal{N}^{(K)}|$ can drastically increase depending on the target nodes
- **If $|\mathcal{N}^{(K)}| > n_c$, MonTi selects top-$n_c$ candidate nodes with $\mathcal{J}(\cdot)$**
  - $\mathcal{J}(G, v) = \mathrm{MLP}\big(\sigma([\mathbf{q}_v \parallel \mathbf{m}_v \parallel \mathbf{h}_v \parallel \mathbf{h}_\mathcal{T}])\big) \in \mathbb{R}$
    - $\mathbf{q}_v = \mathrm{MLP}(\mathbf{x}_v) \in \mathbb{R}^{D_H}$ and $\mathbf{m}_v = \mathrm{MLP}([d_v \parallel \beta_v]) \in \mathbb{R}^{D_H}$
      - d_v is the degree of node v and β_v is the number of target nodes directly connected to node v
    - $\mathbf{h}_v \in \mathbb{R}^{D_H}$ is a representation of node v computed by **a pretrained surrogate GNN** and $\mathbf{h}_\mathcal{T} = \mathrm{READOUT}(\mathbf{h}_t | t \in \mathcal{T}) \in \mathbb{R}^{D_H}$
- **Otherwise, all nodes in $\mathcal{N}^{(K)}$ are considered** as candidate nodes

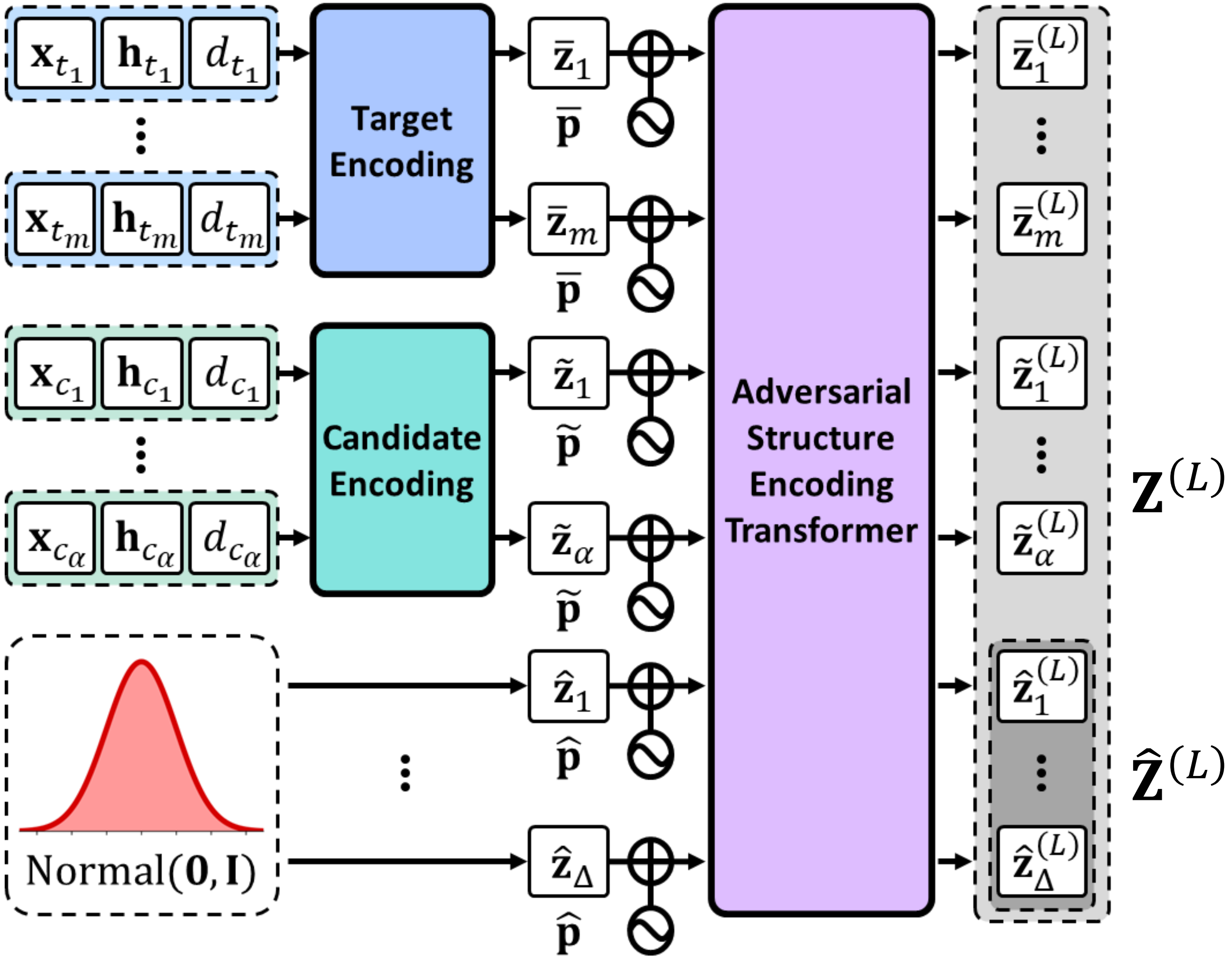
03 Adversarial Structure Encoding



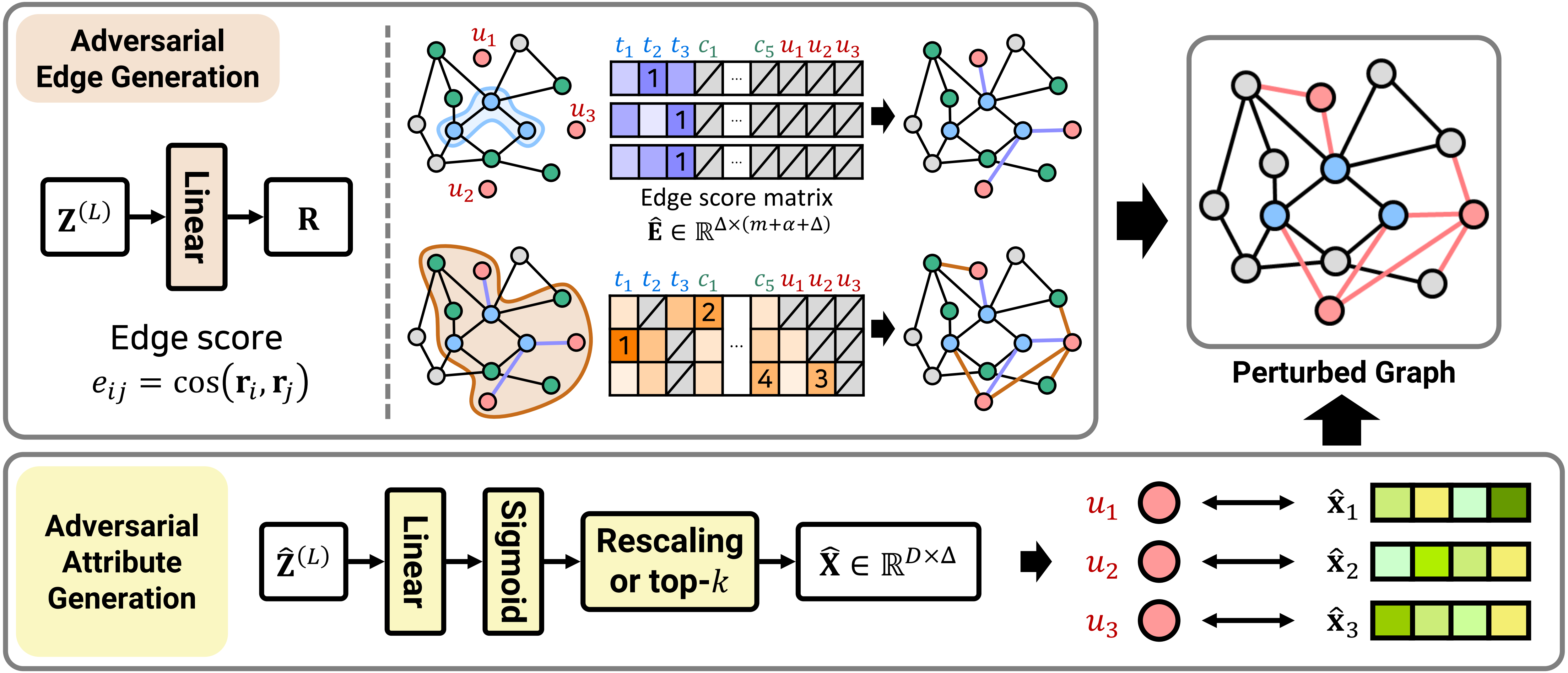
Target Encoding



Candidate Encoding



03 One-Time Graph Injection



03 Training of MonTi

Straight-through Gumbel-top-k

- To solve the optimization problems of **discrete selection in MonTi**, we adopt the **Gumbel-Top-k** technique coupled with the **straight-through** estimator
  - Candidate Selection, Adversarial Edge Generation, Adversarial Attribute Generation for discrete attributes

Loss function

- Following the previous works in graph injection attacks define the loss function based on **C&W loss**

$$\min_{G'} \mathcal{L}\left(f_{\theta^*}, G', \mathcal{T}\right) = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \max\left(s'_{t,1} - s'_{t,0}, 0\right) \text{ where } \mathbf{s}'_t = f_{\theta^*}(G', t) \in \mathbb{R}^2$$

- Focus on **increasing normal scores** and decreasing fraud scores of target nodes to align with our scenarios.
- The loss is **calculated using a surrogate model** (black-box attack)

04 Experiments

Datasets

- Create **3 real-world datasets for multi-target graph injection attacks**
  - GossipCop-S, YelpChi, LifeIns
- Create the training, validation, and test **target sets with fraud nodes** belonging to each split
- **Each target set represents a fraud gang** organized based on **metadata or relations** in each dataset
  - **GossipCop-S**: **Fake news articles** tweeted by **the same user**
  - **YelpChi**: **Fake reviews** of **the same user** or the fake reviews for **the same product within the same month**
  - **LifeIns**: **Fraudulent insurance claims** grouped based on **relationships predefined by domain experts**

	$\|\mathcal{V}\|$	#Frauds	#Target Sets	$\|\mathcal{E}\|$	D	Feature Type
GossipCop-S	16,488	3,898	2,438	3,865,058	768	Continuous
YelpChi	45,900	6,656	1,435	3,846,910	32	Continuous
LifeIns	122,792	1,264	380	912,833	1,611	Discrete

04 Experiments

Surrogate and Victim Models

- **Vanilla GNNs:** GCN (ICLR 2017), GraphSAGE (NIPS 2017), GAT (ICLR 2018)
- **GNN-based fraud detectors:** CARE-GNN (CIKM 2020), PC-GNN (TheWebConf 2021), GAGA (TheWebConf 2023)
- Train all the methods with two different initialization seeds
  - The models initialized with the **first seed** serve as **surrogate models** and **the others** are employed as **victim models**

Attack Baselines

- **Black-box graph injection evasion attack** methods:
  - G-NIA (CIKM 2021), TDGIA (KDD 2021), Cluster Attack (IJCAI 2022), G^2A2C (AAAI 2023)

Evaluation Metric

- **Average misclassification rates** (%) of all target sets weighted by their sizes

Budgets

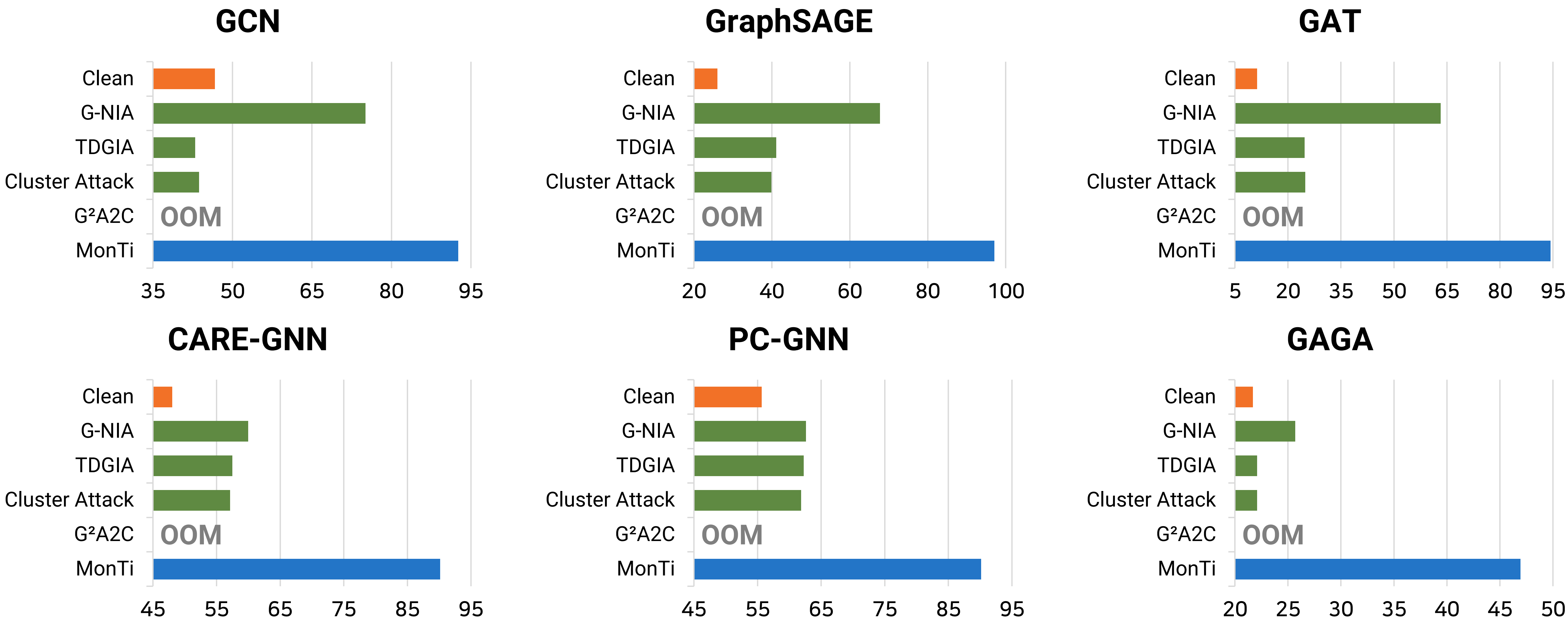
- Due to the **diverse sizes and substructures of target sets**, node and edge budgets should be allocated according to the characteristics of each target set.
- **Impose limits on the budgets** since excessively large budgets can lead to highly noticeable and easy attacks

- **Node budget:** $\Delta = \max(\lfloor \rho \cdot \min(B, \bar{B}) + 0.5 \rfloor, 1)$ where ρ **is a parameter to control node budget**
  - $B \coloneqq \left|\mathcal{N}^{(1)} \cup \mathcal{T}\right|$, $\bar{B}$ is the average value of B across all target sets within the dataset
- **Edge budget:** $\eta = \Delta \cdot \max\left(\left\lfloor \min\left(d_{\mathcal{T}}, \xi \cdot \bar{d}\right) + 0.5 \right\rfloor, 1\right)$ where ξ **is a parameter to control edge budget**
  - $d_{\mathcal{T}}$ is the average node degree of the target set, $\bar{d}$ is the average node degree of all nodes in the graph

- We set $\rho = 0.05, \xi = 0.1$ for GossipCop-S, $\rho = 0.05, \xi = 0.5$ for YelpChi, and $\rho = 0.2, \xi = 0.5$ for LifeIns

04 Multi-Target Attack Performance

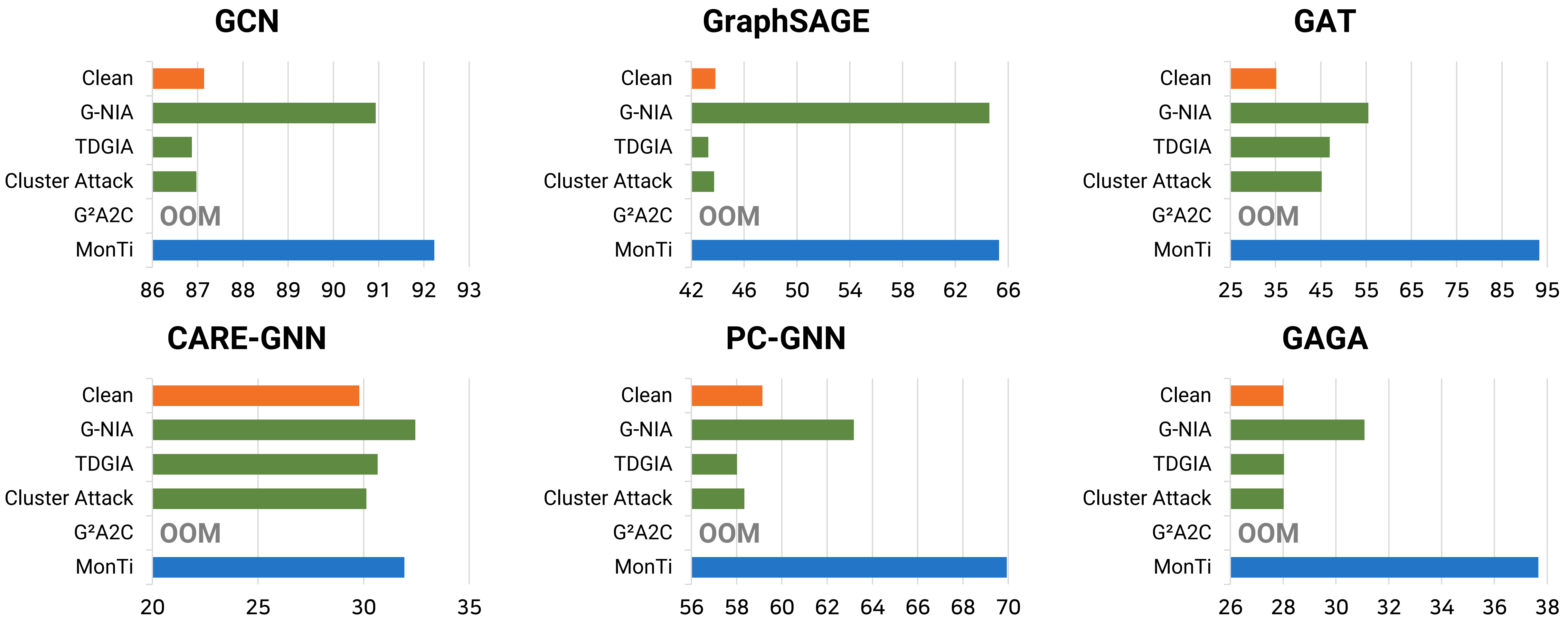
Misclassification rates (%) on GossipCop-S when the types of surrogate and victim models are the same



* OOM: Out of Memory Error

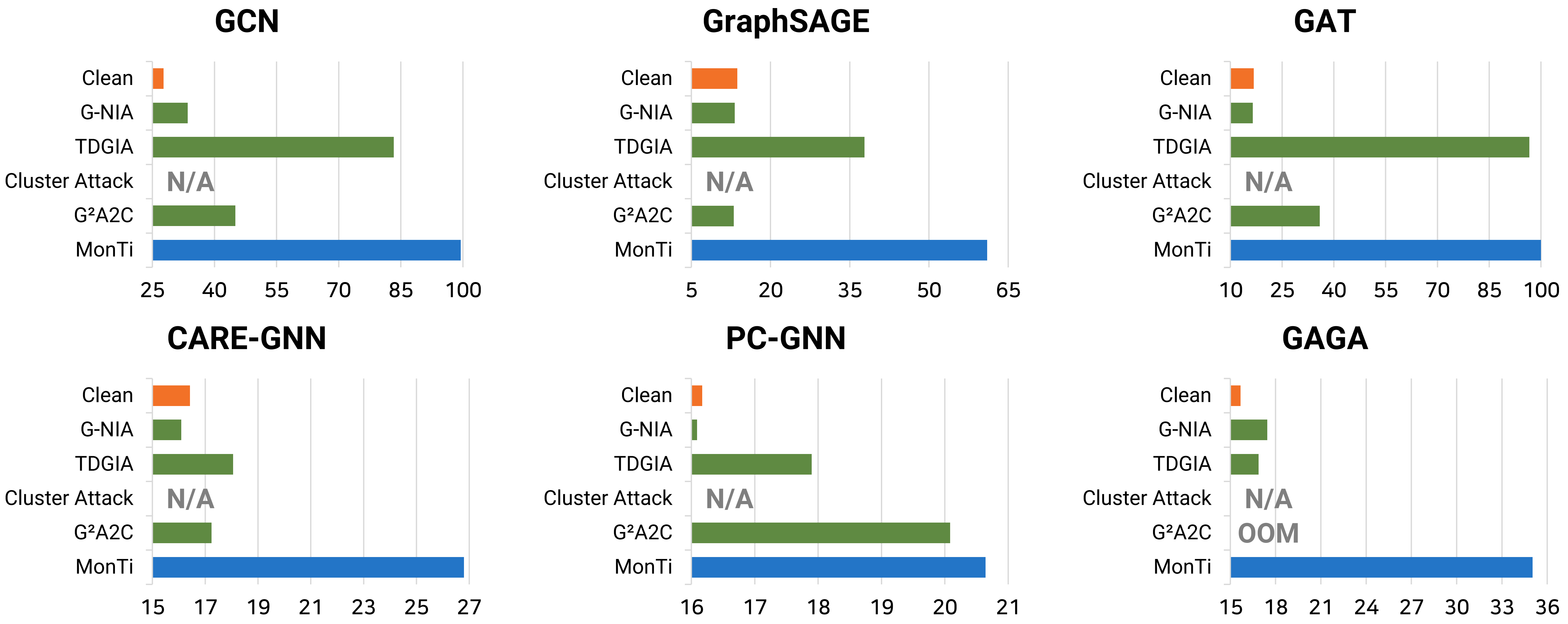
04 Multi-Target Attack Performance

Misclassification rates (%) on YelpChi when the types of surrogate and victim models are the same



04 Multi-Target Attack Performance

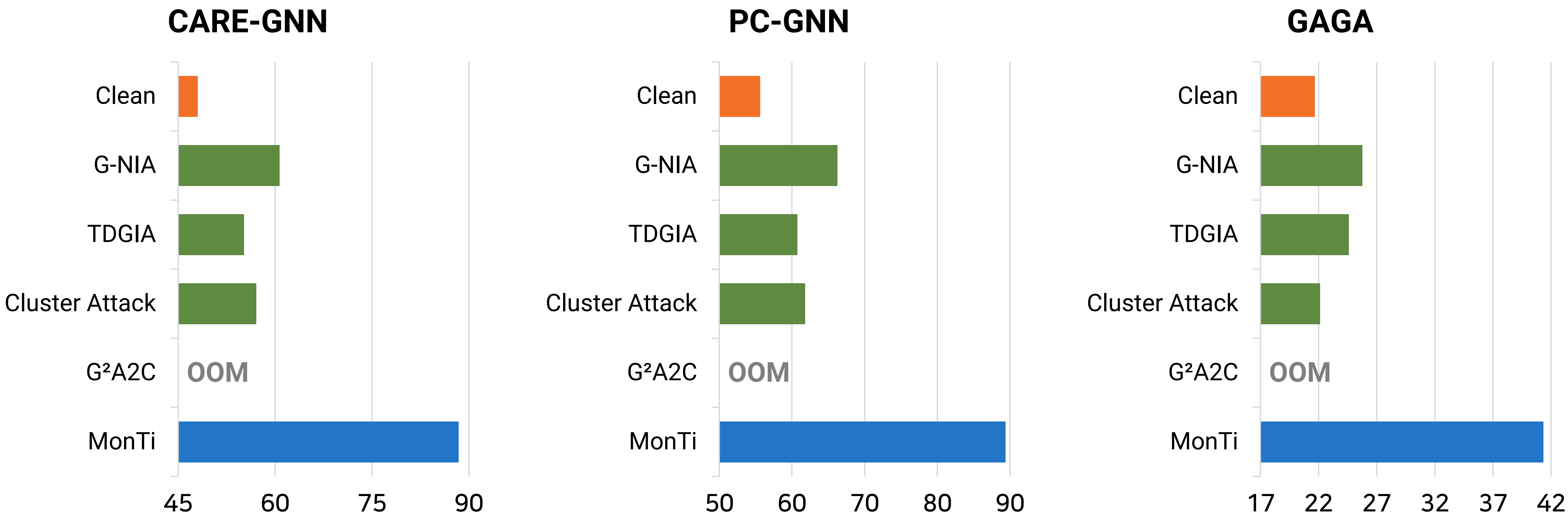
Misclassification rates (%) on LifeIns when the types of surrogate and victim models are the same



* N/A: Not Completed in 5 days

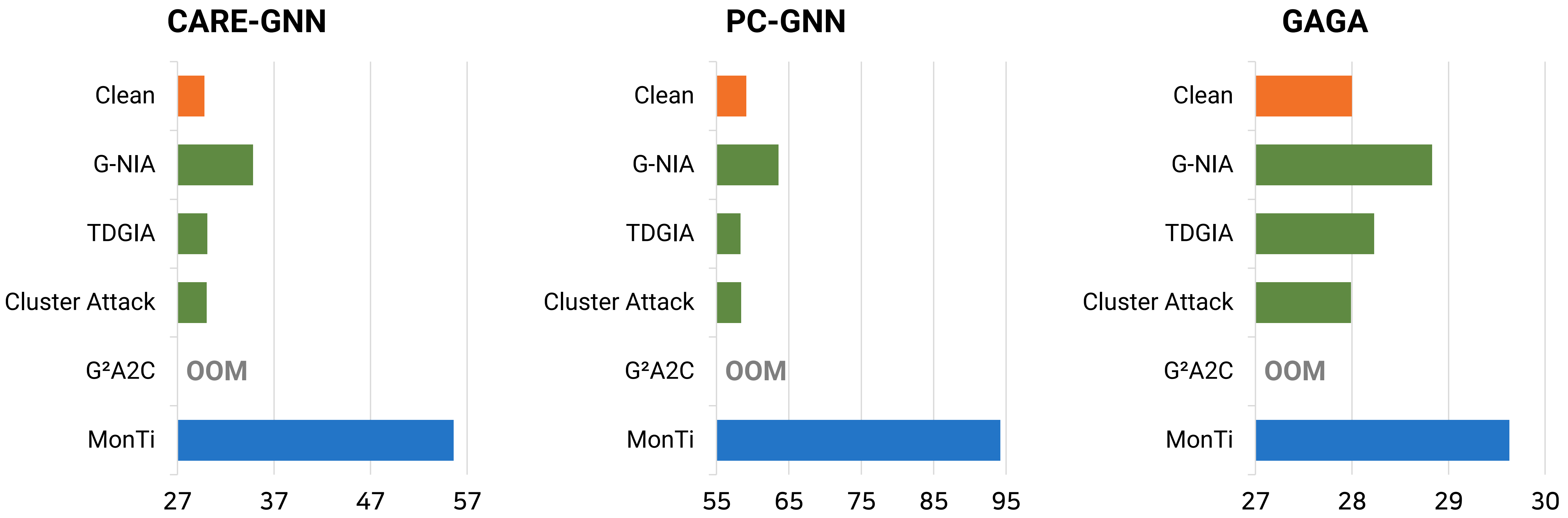
04 Multi-Target Attack Performance

Misclassification rates (%) on GossipCop-S when GCN is the surrogate model



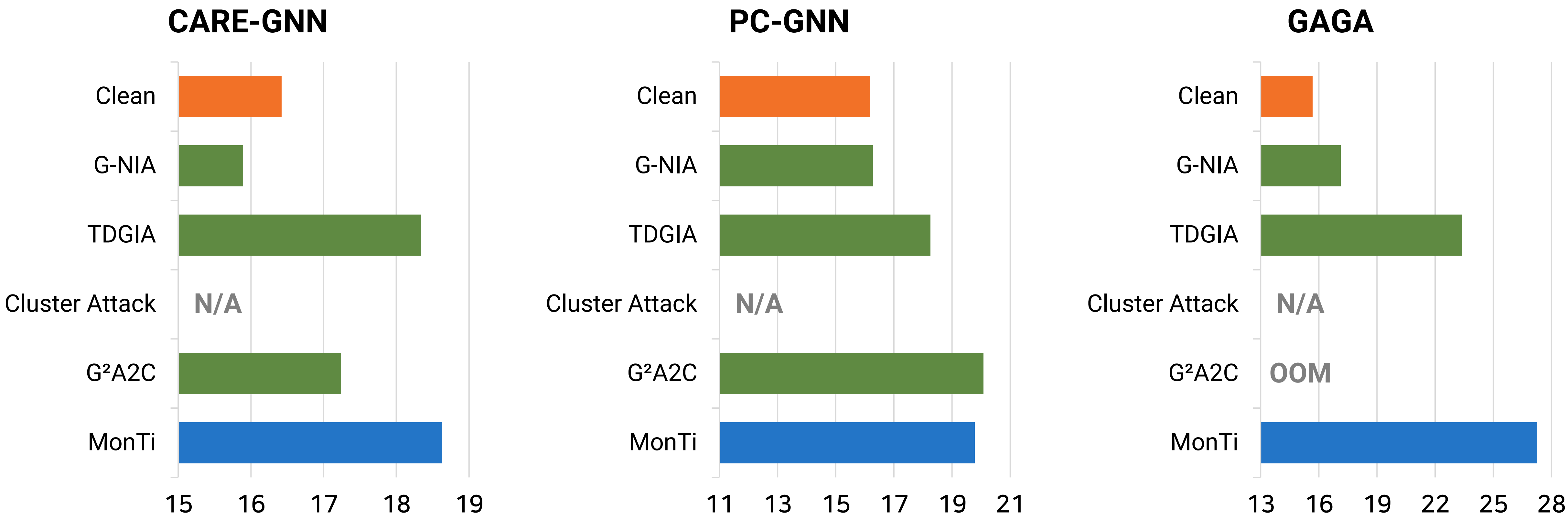
04 Multi-Target Attack Performance

Misclassification rates (%) on YelpChi when GCN is the surrogate model



04 Multi-Target Attack Performance

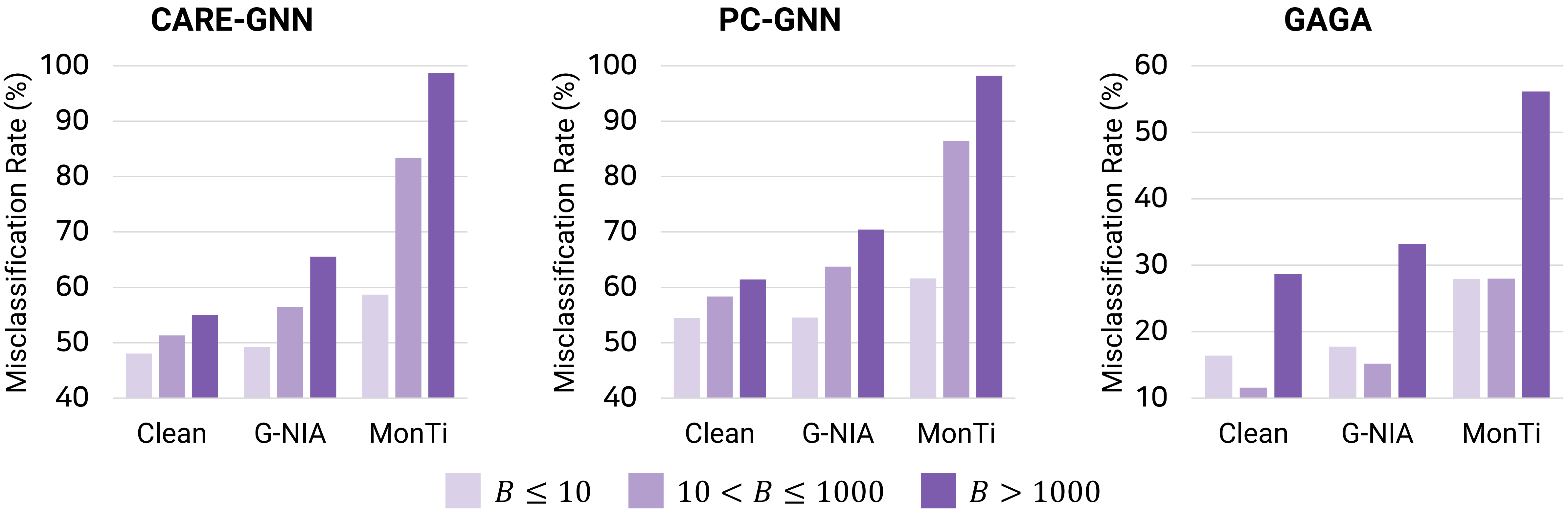
Misclassification rates (%) on LifeIns when GCN is the surrogate model



Case Study: Effects of the Size of Fraud Gangs

Categorize target sets into three groups based on $B \coloneqq \left|\mathcal{N}^{(1)} \cup \mathcal{T}\right|$

- On **GossipCop-S** using **GCN as the surrogate model**
- B reflects the size of the fraud gang

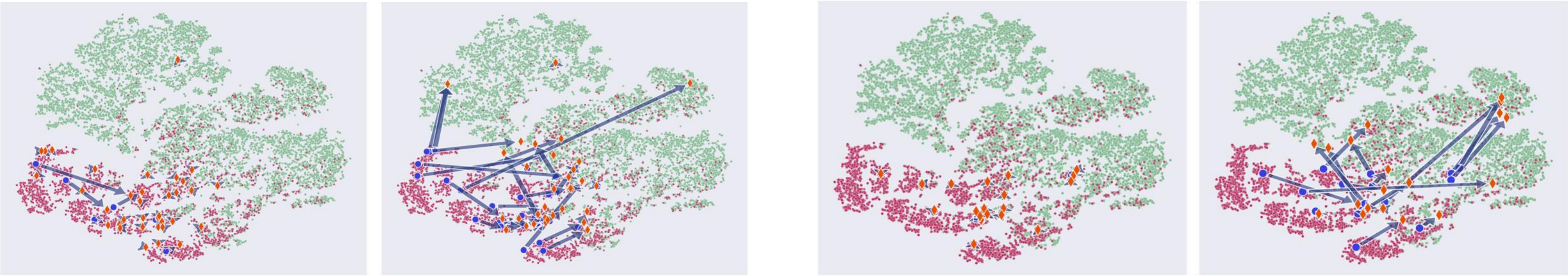


Case Study: Effects of the Size of Fraud Gangs

Visualize the latent representations of target nodes computed by **GAGA before and after the attack**

- On **GossipCop-S** using **GCN as the surrogate model** and focusing target sets with $B > 1000$
- MonTi significantly shifts the representations from the fraud to the benign area

G-NIA / GAGA / GossipCop-S | MonTi / GAGA / GossipCop-S | G-NIA / GAGA / GossipCop-S | MonTi / GAGA / GossipCop-S



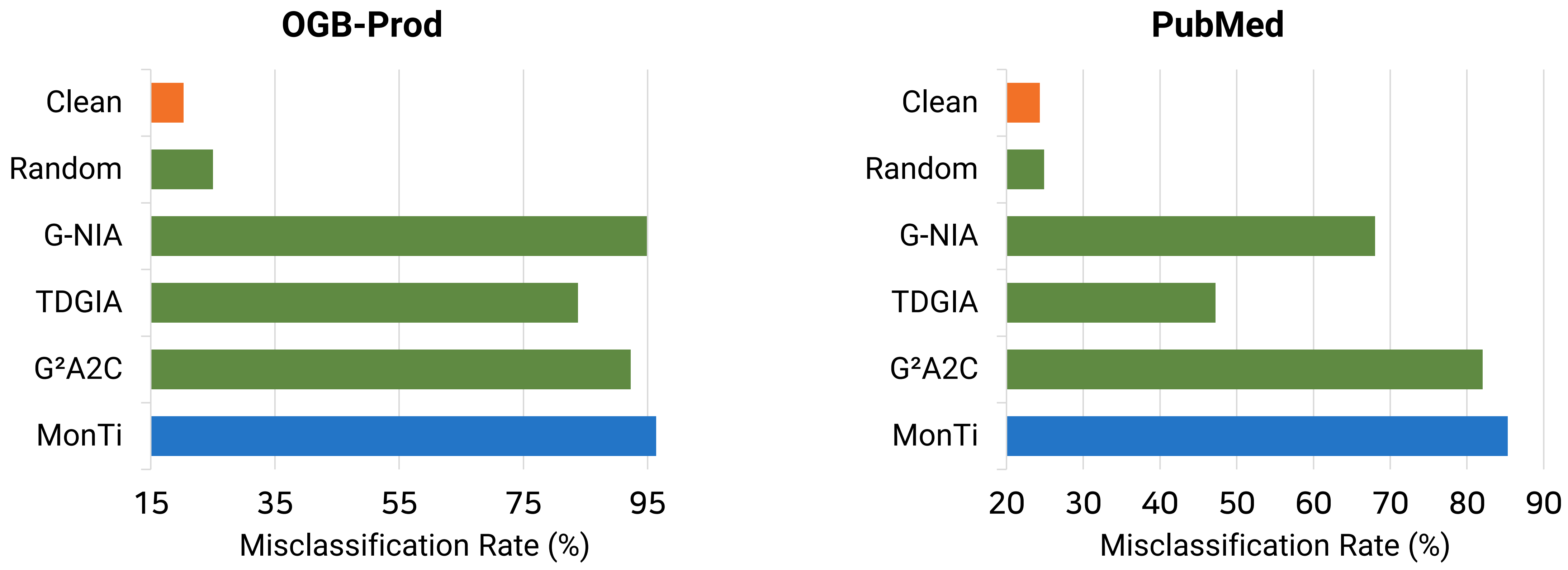
Misclf. Rate: 25.7% → 28.6% $|\mathcal{T}| = 35, B = 3645$ | Misclf. Rate: 25.7% → 68.6% $|\mathcal{T}| = 35, B = 3645$ | Misclf. Rate: 42.1% → 42.1% $|\mathcal{T}| = 19, B = 7423$ | Misclf. Rate: 42.1% → 84.2% $|\mathcal{T}| = 19, B = 7423$

Benign Fraud Target Attacked Target

04 Single-Target Attack Performance

On OGB-Prod and PubMed with GCN as the surrogate and victim models $(\Delta = 1, \eta = 1)$

- Despite not being specifically designed for single-target attacks, MonTi outperforms all baselines



05 Conclusion

- **Multi-target graph injection attacks** against **GNN-based fraud detectors** with practical scenarios
  - First study to explore adversarial **attacks against GNN-based fraud detectors** and **graph injection attacks for multiple target nodes formed by fraud gangs**

- **Proposed method MonTi** achieves flexible and efficient attacks by **adaptively allocating the degree budget** for each attack node and injecting all attack nodes **at once**

- MonTi effectively captures **interdependencies between node attributes and edges**, as well as **interactions within target nodes** and **among attack nodes**

- MonTi significantly outperforms state-of-the-art graph injection attack methods

Thank You!

▲ GitHub

▲ BDILab

Our datasets and codes are available at:
https://github.com/bdi-lab/MonTi

You can find us at:
{cjh0507, heehyeon, jjwhang}@kaist.ac.kr
https://bdi-lab.kaist.ac.kr

KAIST BDILab
BIG DATA INTELLIGENCE