A Benchmark of Graph Augmentations for Contrastive Learning-Based Network Attack Detection with Graph Neural Networks

Tristan Bilot[1,2,4], Nour El Madhoun[1,2,3], Khaldoun Al Agha[1] and Anis Zouaoui[4]

[1] LISN, Université Paris-Saclay, FR
[2] LISITE, ISEP, FR
[3] LIP6, Sorbonne Université, FR
[4] Iriguard, FR

requests: tristan.bilot@universite-paris-saclay.fr

Motivation

Acquiring labeled datasets in the cybersecurity domain is challenging. Consequently, efforts are directed towards learning representations directly from data using self-supervised approaches. In this work, we focus on contrastive methods that aim to learn embeddings from the graph structure along with integrated features, using various graph augmentations. Our goal is to benchmark 10 augmentation techniques and provide more efficient augmentations for network data. We systematically evaluate 100 pairs of positive and negative graphs and present our findings in a heatmap, highlighting the best-performing techniques.

Goal

In the Graph Contrastive Learning (GCL) setting, we aim to optimize a neural network f_θ such that:

$$f_\theta(G) \approx f_\theta(G^+)$$
$$f_\theta(G) \neq f_\theta(G^-)$$

where G is the anchor graph, and G^+ and G^- represent the positive and negative graph, respectively. In other words, we want to obtain similar embeddings for the anchor and positive graphs, and different embeddings for the negative graph.
In this work, we benchmark multiple augmentation functions responsible of the generation of graphs G^+ and G^-. Effective augmentation functions have a direct impact on the model's performance.

CSNet 2023, Montréal

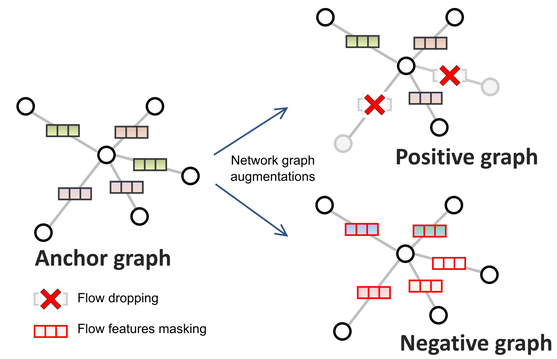
How various graph augmentations influence contrastive learning in GNNs?

A benchmark on a network attack detection scenario

Graph augmentations

In contrastive learning, each graph in the dataset (called **anchor** graph) is augmented into 2 altered versions, a **positive** and a **negative** graph.

The model then learns to **maximize agreement** between the anchor graph and positive graph, while **minimizing agreement** with the negative graph. This allows the model to learn **embeddings** in a self-supervised way.



Experiments & results

On 2 network attack datasets, we benchmarked **10 graph augmentations** on both positive and negative graphs, for a total of 100 augmentation pairs.

Our experiments show that various augmentations improve the performance of baseline augmentations used in contrastive learning methods. Improvements are justified by a **1.8%** and **2.2% F1-score increase** on both datasets.

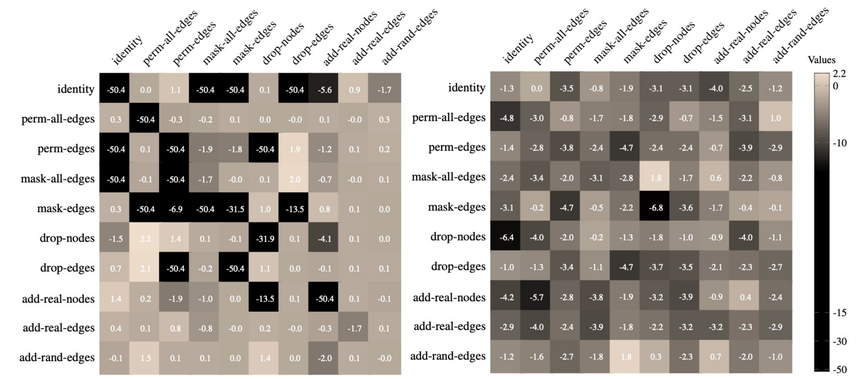


Fig. 2. Heatmaps representing the macro F1-score gain (%) between each pair of augmentations and the baseline method using the *identity/perm-all-edges* augmentation. Each row indicates a positive augmentation, whereas each column represents a negative augmentation. Left: NF-CSE-CIC-IDS2018-v2 dataset; Right: NF-UNSW-NB15-v2 dataset.

Benchmarked augmentations

The 10 benchmarked augmentation functions can be divided into 3 categories:

Attributive augmentations

- ***perm-all-edges***: randomly permutes all edge features
- ***perm-edges***: randomly permutes N% of edge features
- ***mask-all-edges***: replaces all edge features by random features generated using Gaussian noise
- ***mask-edges***: replaces N% of the edge features by random features generated using Gaussian noise

Topological augmentations

- ***drop-nodes***: randomly removes N% of the nodes along with all the connected edges
- ***drop-edges***: randomly removes N% of the edges

Hybrid augmentations

- ***add-real-nodes***: creates N% of new nodes and generates k incoming edges and d outgoing edges with randomly-peeked edge features, where k and d denote the mean of nodes' in-degree and out-degree, respectively
- ***add-real-edges***: creates N% of new edges with randomly-peeked edge features, using random end nodes
- ***add-rand-edges***: creates N% of new edges with features generated by Gaussian noise, using random end nodes

The 10th augmentation, ***identity***, simply returns the same graph given as input.

Setup

- The augmentation pair usually employed in GCL is ***identity/perm-all-edges****. This pair is thus the baseline used as comparison on both heatmaps.
- In our experiments, the E-GraphSAGE [1] model is used as encoder, Deep Graph Infomax (DGI) [2] is used as GCL module and an Isolation Forest as classifier. This architecture has been also used in Anomal-E [3].
- We found that N=30% was a good value for the sampling-based augmentation functions.
- All edges contain edge features, and performance is measured on the edge classification task.

Conclusion

From our experiments, ***attributive/topological*** and ***topological/attributive*** augmentations outperform others, with the top four pairs showing a 1.9% to 2.2% F1 gain. Notably, the ***mask-all-edges/drop-edges*** pair randomizes edge features in the positive graph but retains its topology, yielding strong results. However, augmentations effective on one dataset might not work as well on another. Exploring the link between augmentations and graph structures is a direction for future research

[1] Wai Weng Lo, Siamak Layeghy, Mohanad Sarhan, Marcus Gallagher, and Marius Portmann. E-graphsage: A graph neural network based intrusion detection system for iot. NOMS 2022-2022 IEEE/IFIP
[2] Petar Velickovic, William Fedus, William L Hamilton, Pietro Lio`, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. ICLR (Poster), 2(3):4, 2019.
[3] Evan Caville, Wai Weng Lo, Siamak Layeghy, and Marius Portmann. Anomal-e: A self-supervised network intrusion detection system based on graph neural networks. Knowledge-Based Systems, 258:110030, 2022.

** : to identify a pair, we use the notation **positive/negative***