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

Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, Anis Zouaoui

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

Tristan Bilot*†‡, Nour El Madhoun†‡§, Khaldoun Al Agha†, Anis Zouaoui*

* Iriguard, 5 Rue Bellini, 92800, Puteaux
† Université Paris-Saclay, CNRS, Laboratoire Interdisciplinaire des Sciences du Numérique, 91190, Gif-sur-Yvette, France
‡ LISITE Laboratory, ISEP, 10 Rue de Vanves, Issy-les-Moulineaux, 92130, France
§ Sorbonne Université, CNRS, LIP6, 4 place Jussieu 75005 Paris, France
Email: {tristan.bilot,nour.el-madhoun}@universite-paris-saclay.fr; alagha@lisn.fr;
{t.bilot,a.zouaoui}@iriguard.com; {tristan.bilot, nour.el-madhoun}@isep.fr

***Abstract*—Graph Neural Networks (GNNs) have recently emerged as powerful tools for detecting network attacks, due to their ability to capture complex relationships between hosts. However, 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 study, we focus on contrastive methods that aim to maximize agreement between the original graph and positive graph augmentations, while minimizing agreement with negative 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 table, highlighting the best-performing techniques. In particular, the experiments demonstrate that leveraging topological and attributive augmentations in the positive and negative graph generally improves performance, with up to 1.8% and 2.2% improvement in F1-score on two different datasets. The analysis further showcases the intrinsic connection between the performance of graph augmentations and the underlying data, highlighting the need for careful prior selection to achieve optimal results.**

***Index Terms*—Attack Detection, Network Security, Contrastive Learning, Self-Supervised Learning, Graph Neural Networks.**

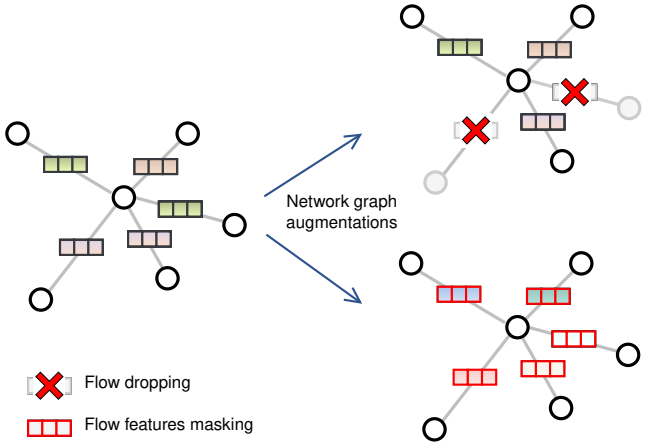

Fig. 1. Graph augmentations play a key role in enhancing self-supervised graph contrastive learning methods for network-based attack detection. This figure illustrates flow dropping and flow features masking augmentations, which are among several other augmentations presented in this paper.

I. Introduction

Attack detection plays a crucial role in ensuring the security and integrity of computer networks. With the growing complexity and sophistication of attacks, it has become imperative to develop powerful tools and techniques to accurately identify and mitigate network threats. Graph Neural Networks (GNNs) have emerged as promising solutions due to their ability to capture complex relationships and dependencies between network entities [1]. Moreover, the interconnected nature of networks allows them to be naturally represented as graphs, where nodes represent network entities and edges capture the relationships between them. This inherent graph structure makes it well-suited to apply GNNs as they excel in capturing and modeling dependencies within graph-structured data. By leveraging the connectivity and topology of network data, GNNs have the potential to uncover hidden patterns and underlying characteristics that are essential for effective detection of network attacks.

Traditionally, network attack detection methods heavily rely on labeled datasets, where expert domain knowledge is used to annotate attack cases. However, in the cybersecurity domain, obtaining such labeled datasets is often difficult and time-consuming. This limitation has led researchers to explore alternative approaches that can learn representations directly from raw network data, without the need for explicit labeling. Self-supervised learning has received considerable attention as an effective approach in this respect, enabling GNNs to acquire meaningful representations by taking advantage of the inherent structure and patterns present in the network data. The aim of self-supervised learning is to design pretext tasks that can capture the underlying

structure of the data [2]. By formulating the learning task appropriately, the network can learn representations that encode crucial information about the network topology, relationship between hosts and potential attack patterns. In recent years, contrastive learning has emerged as a promising self-supervised learning framework. Contrastive learning aims to maximize agreement between positive augmentations (perturbed versions) of the original graph and minimize agreement with negative augmentations, thus encouraging the network to capture relevant features and distinguish between normal and abnormal behavior.

In this study, we focus on benchmarking various graph augmentation techniques, such as those illustrated in Fig. 1, within the contrastive learning framework for network attack detection using GNNs. Our goal is to evaluate the performance and effectiveness of ten augmentation strategies in enhancing the discriminative power of self-supervised representations. By conducting a comprehensive evaluation using two real-world network attack datasets, we aim to provide insight into the strengths and weaknesses of different augmentation techniques. This analysis will help to identify more effective graph augmentations and improve the accuracy and robustness of network attack detection systems.

The remainder of this paper is organized as follows: Section 2 provides an overview of graph contrastive learning and a description of the main principles behind graph augmentations. Section 3 presents the experimental setup, along with the different augmentations benchmarked in this paper and their corresponding results. Section 4 concludes the paper and outlines the main findings of this study.

This research contributes to the advancement of GNN-based network attack detection by exploring the potential of graph augmentations in the contrastive learning paradigm, an area that remains underexplored, particularly in the domain of network security. While many existing works in this field rely on the use of the anchor graph as a positive graph, our study provides evidence and valuable insights into the effectiveness of different augmentation pairs. By doing so, we aim to improve the development of detection systems based on contrastive learning and GNNs.

II. Background

A. Graph Contrastive Learning

Graph contrastive learning is a specific formulation of self-supervised learning that aims to maximize agreement between positive graph augmentations and minimize agreement with negative graph augmentations [2], [3]. Let G and G^+ represent the original graph and its positive augmentation, respectively, and G^- represent a negative augmentation. The goal is to learn a representation function f_θ such as a GNN, that maps an input graph to a meaningful latent representation. This function satisfies:

$$f_\theta(G) \approx f_\theta(G^+), \tag{1}$$

$$f_\theta(G) \not\approx f_\theta(G^-), \tag{2}$$

where $\approx$ denotes high similarity between the original graph representation and its positive augmentation, and $\not\approx$ denotes low similarity between the original and negative graph. The agreement between the resulting graph representations is then estimated using a discriminator function g_ϕ such as a Multi-Layer-Perceptron (MLP), which projects the representations to another latent space where the loss will be calculated. Specifically, the functions f_θ and g_ϕ can be formalized as encoder and decoder functions, respectively parameterized by weights θ and ϕ. The weights are updated by minimizing a contrastive loss function $\mathcal{L}_{ssl}$, with the aim of generating similar embeddings for positive graphs and dissimilar embeddings for negative graphs. The contrastive learning framework can be summarized as:

$$\theta^*, \phi^* = \arg\min_{\theta,\phi} \mathcal{L}_{ssl}\left(g_\phi\left(f_\theta\left(G^+\right), f_\theta\left(G^-\right)\right)\right), \tag{3}$$

where θ^* and ϕ^* denote the newly updated weights. The trained model resulting from Eq. 3 can ultimately produce graph representations that are valuable for a wide range of supervised or unsupervised downstream tasks, including node/edge/graph classification, clustering, and anomaly detection.

B. Graph Augmentations

Graph augmentations play a crucial role in graph contrastive learning by introducing variations in the original graph (also called an anchor graph), enhancing the model's ability to capture important graph properties and improve discriminability. Augmentations are generally applied to generate both positive and negative samples, allowing the model to learn to discriminate between them effectively. The choice of graph augmentations in contrastive methods is crucial for capturing diverse features. Different augmentations introduce variations in the graph structure and attributes, enabling the model to learn robust representations. This work evaluates the main types of graph augmentations, which are explained in more detail in section III.

III. Experiments

A. Setup

Our experiments are conducted using the Anomal-E detection system [4], which employs contrastive learning with GNNs for network-based intrusion detection. Anomal-E is based on Deep Graph Infomax (DGI) [5], a contrastive method specifically designed for self-supervised representation learning with GNNs. Initially, a GNN encoder computes node representations for both the anchor graph and a negatively augmented graph. The representations of the anchor graph are then passed through a readout function to generate a graph summary that captures the global information embedded within the graph. The representations are learned by optimizing a binary classification objective using binary cross-entropy, with the goal of distinguishing between the anchor graph and the negative graph. This method effectively maximizes the mutual information between local and global

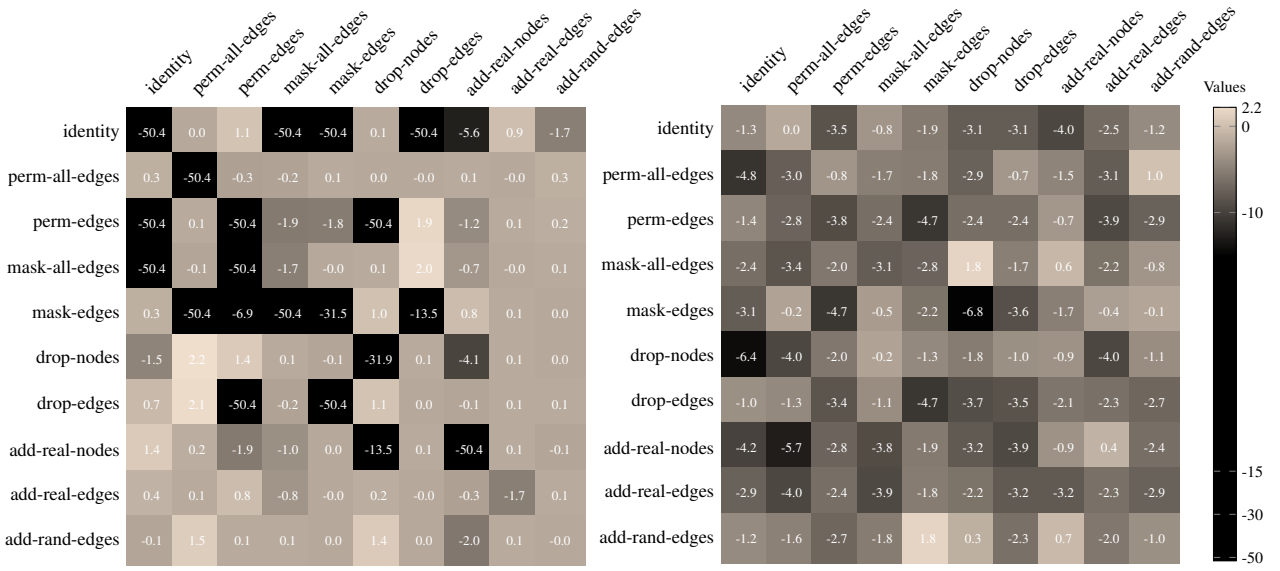

Fig. 2. Heatmaps representing the macro F1-score gain (%) between each pair of augmentations and the baseline method using the *identity*/*perm-all-edges* augmentation. Each row indicates a positive augmentation, whereas each column represents a negative augmentation. Left: NF-CSE-CIC-IDS2018-v2 dataset;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

representations, enabling the encoder to learn localized and global semantic information more effectively.

We adopt the same datasets, configurations and preprocessing steps outlined in Anomal-E, ensuring consistency and facilitating direct comparisons with the graph augmentation leveraged in the paper. Specifically, Anomal-E uses random edge permutation as augmentation for negative graphs, and leverage the anchor graph as positive sample (denoted here as the identity augmentation), similarly to the original DGI model. The first benchmark dataset, NF-CSE-CIC-IDS2018-v2 [6], contains 18,893,708 flows with around 12% attack samples, whereas the second dataset NF-UNSW-NB15-v2 [6] contains 2,295,222 flows with 4% attack samples. Both datasets have been standardized using the Netflow format and 43 network flow features are available and used here as edge features in the homogeneous graph. All experiments were conducted in parallel on 6 NVIDIA Tesla V100 GPUs.

B. Augmentations

In the following experiments, we propose to benchmark multiple augmentations for both positive and negative graphs, categorized in three types [2], in an attempt to improve the augmentations used in DGI and Anomal-E.

Attributive Augmentations. Attributive augmentations focus on manipulating the node and edge attributes. These augmentations involve operations such as attribute masking, attribute perturbation, or attribute shuffling. By altering the features in the graph, the model can learn to extract robust and discriminative representations that capture the underlying patterns and relationships among attributes. In these experiments, we used the following attributive augmentations:

- *perm-all-edges*: randomly permutes all edge features.
- *perm-edges*: randomly permutes $N\%$ of the 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. Topological augmentations involve modifying the graph's connectivity patterns. This technique includes operations such as edge perturbation, edge addition or deletion, or graph rewiring. By introducing variations in the graph's topology, the model learns to discern different edge configurations and gain a better understanding of the graph structure. This work benchmarks the following 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. Hybrid augmentations combine both attributive and topological techniques to create diverse alterations in the graph. By leveraging both attributive and topological aspects, hybrid augmentations enable the model to capture a wide range of features and relationships, which may lead to more effective representations. In the context of this work, we implemented the following 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.

For each augmentation requiring the sampling parameter N, multiple simulations have been performed using various values of N. The simulations conclude that a better performance is reached with $N = 30\%$.

Using these augmentations, we aim to craft many positive and negative edges, leveraged during the self-supervised training of the model. Afterwards, the trained model should learn how to distinguish between positive and negative edges, by directly clustering them in the embedding space.

C. Results

The two heatmaps presented in Fig. 2 show the experimental results on both presented datasets. Each value indicates the macro F1-score gain between a *positive*/*negative* pair of augmentations presented previously, and the baseline augmentation used in Anomal-E, namely the *identity*/*perm-all-edges* augmentation (located in the heatmaps at the first row and second column). The gain specifically measures the percentage of increase in F1-score, compared to the baseline method, and is computed using a mean over 5 experiments. For sampling-based augmentations, N is set to 30, and the learning rate is respectively set to 0.003 and 0.002 for the NF-CSE-CIC-IDS2018-v2 and NF-UNSW-NB15-v2 datasets. The E-GraphSAGE model [7] is used as GNN encoder, and an Isolation Forest (IF) classifier [8] is used as downstream algorithm for classifying the edge embeddings. The parameters of the IF are grid-searched using the same techniques employed in Anomal-E. The gain values were calculated using the F1-scores achieved by the Anomal-E baseline on both datasets. Specifically, the F1-scores obtained were 94.47% and 91.02% for each datasets.

On the NF-CSE-CIC-IDS2018-v2 dataset, the experiments show that best performance is reached using the *drop-nodes*/*perm-all-edges* and *drop-edges*/*perm-all-edges* pairs, with a 2.2% and 2.1% F1-score gain, respectively. This means that better embeddings are produced when the discriminator of DGI maximizes the similarity between the anchor graph and a positive graph altered by topological augmentations. Generally, *attributive*/*topological* and *topological*/*attributive* augmentations tend to perform better than all other pairs of augmentations, as shown by the 4 best pairs ranging from 1.9% and 2.2% gain. It is also worth noting interesting pairs, such as *mask-all-edges*/*drop-edges*, which achieves impressive results by randomizing all edge features in the positive graph while preserving the graph topology. Nonetheless, we also notice multiple pairs with large negative gains, which are mostly characterized by either too simple or too hard augmentations to learn [9].

Using the NF-UNSW-NB15-v2 dataset, we observe a notable contrast in the data compared to the previous dataset, characterized by smoother gains, mostly negatives. This implies that the baseline *identity*/*perm-all-edges* augmentation performs relatively well on this network graph compared to most other augmentation pairs. Nonetheless, we identify three augmentation pairs that exhibit superior performance, surpassing the baseline by at least a 1% gain, with a best 1.8% gain using *mask-all-edges*/*drop-nodes* and *add-rand-edges*/*mask-edges* augmentations. These findings reaffirm that combining topological and attributive augmentations can lead to enhanced results.

On both datasets, it becomes evident that selecting the identity augmentation as the positive graph is often suboptimal. This indicates that prevailing contrastive approaches with GNNs should take into account the evaluation of multiple positive augmentations in order to maximize performance. Indeed, the performance of any augmentation technique is intrinsically linked to the underlying data, making the automatic selection of augmentations challenging without benchmarking multiple augmentation pairs. Furthermore, conducting additional experiments on diverse datasets could aid in the discovery of more general and effective augmentations for network datasets.

IV. Conclusion

In this study, we explored the impact of various graph augmentation techniques on the performance of network attack detection systems based on contrastive learning and GNNs. Through a comprehensive analysis of ten different augmentations, we assessed their effects on the capture of both topological and attributive information. Our experimental findings demonstrated that incorporating topological and attributive augmentations in both positive and negative graphs consistently outperforms baseline techniques that solely rely on the anchor graph as the positive graph. Our investigation underscored the importance of graph augmentation strategies as an additional hyperparameter in training GNN-based contrastive approaches for detection systems, emphasizing its role in improving detection accuracy.

References

[1] Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, and Anis Zouaoui. Graph neural networks for intrusion detection: A survey. *IEEE Access*, 2023.

[2] Yixin Liu, Ming Jin, Shirui Pan, Chuan Zhou, Yu Zheng, Feng Xia, and S Yu Philip. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(6):5879–5900, 2022.

[3] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

[4] 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.

[5] Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *ICLR (Poster)*, 2(3):4, 2019.

[6] Mohanad Sarhan, Siamak Layeghy, and Marius Portmann. Towards a standard feature set for network intrusion detection system datasets. *Mobile networks and applications*, pages 1–14, 2022.

[7] 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 Network Operations and Management Symposium*, pages 1–9, 2022.

[8] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. *2008 eighth ieee international conference on data mining*, pages 413–422, 2008.

[9] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.