Pegah Golchin, Nima Rafiee, Mehrdad Hajizadeh, Ahmad Khalil, Ralf Kundel, Ralf Steinmetz. *SSCL-IDS: Enhancing Generalization of Intrusion Detection with Self-Supervised Contrastive Learning*. To appear in Proceedings of the International Federation for Information Processing (IFIP) Networking Conference, 2024.

SSCL-IDS: Enhancing Generalization of Intrusion Detection with Self-Supervised Contrastive Learning

Pegah Golchin *, Nima Rafiee †, Mehrdad Hajizadeh ‡, Ahmad Khalil *, Ralf Kundel *, Ralf Steinmetz *

Multimedia Communications Lab (KOM), Technical University of Darmstadt, Germany

†*Zalando, Berlin, Germany*

‡*Communication Networks, Technical University of Chemnitz, Germany*

Contact: pegah.golchin@kom.tu-darmstadt.de

***Abstract*—With the increasing diversity and complexity of cyber attacks on computer networks, there is a growing demand for Intrusion Detection Systems (IDS) that can accurately categorize new unknown network flows. Machine learning-based IDS (ML-IDS) offers a potential solution by learning underlying network traffic characteristics. However, ML-IDS encounters performance degradation in predicting the traffic with a different distribution from its training dataset (i.e., new unseen data), especially for attacks that mimic benign (non-attack) traffic (e.g., multi-stage attacks). Diversity in attack types intensifies the lack of labeled attack traffic, which leads to reduced detection performance and generalization capabilities of ML-IDS. The generalization refers to the model's capacity to identify new and unseen samples, even in cases where their distribution deviates from the training data used for the ML-IDS. To address these issues, this paper introduces *SSCL-IDS*, a Self-Supervised Contrastive Learning IDS designed to increase the generalization of ML-IDS. The proposed *SSCL-IDS* is exclusively trained on benign flows, enabling it to acquire a generic representation of benign traffic patterns and reduce the reliance on annotated network traffic datasets. The proposed *SSCL-IDS* demonstrates a substantial improvement in detection and generalization across diverse datasets compared to supervised (over 27%) and unsupervised (over 15%) baselines due to its ability to learn a more effective representation of benign flow attributes. Additionally, by leveraging transfer learning with *SSCL-IDS* as a pretrained model, we achieve AUROC scores surpassing 80% when fine-tuning with less than 20 training samples. Without fine-tuning, the average AUROC score across different datasets resembles random guessing.**

***Index Terms*—Network Intrusion Detection System, Machine Learning, Self-Supervised Learning**

I. Introduction

Nowadays, nearly two-thirds of the global population has access to the Internet, highlighting how deeply intertwined it has become in our daily lives [1]. Nevertheless, this widespread connectivity has also led to a notable increase in cyber attacks, doubling over the past five years, from 2018 to 2023 [1]. A recent study [2] revealed that 80% of security breaches stem from *zero-day* attacks, which are distinguished by their lack of prior information or defenses (i.e., unseen and unknown attacks). These attacks incur an average cost of 1.2 million per incident [2]. This underscores the severity of the threat posed by zero-day attacks and the need for a network Intrusion Detection System (IDS) capable of detecting them.

Traditional signature-based IDSs are restricted to identifying known attacks stored in their databases. Consequently, they exhibit shortcomings in detecting zero-day attacks or any attack that is not stored in their databases [3]. Conversely, anomaly-based IDSs, leveraging Machine Learning (ML) models, can discern statistical patterns in network traffic, enabling the classification of unseen flows with similar (not exact) patterns.

Supervised ML models can be used in IDSs to learn the relationship between statistical features of flows and ground-truth labels (Attack/Benign), which exhibit high detection performance. However, in practical scenarios, the availability of labeled network traffic is often limited, demanding extensive human involvement for labeling, which can be infeasible in certain cases [4]. On the other hand, unsupervised learning models (e.g., autoencoders) can address the network intrusion detection task without annotated data by reconstructing network traffic and learning its abstract features. However, these models are limited to learning features that are relevant to the categorization of the training data, which may restrict their detection capability to flows resembling the training data distribution [5].

In real-world scenarios, the evolving network architecture, varied network management rules, and the availability of new technologies (e.g., Internet of Things) can lead to diversified traffic pattern distributions [6], [7]. Additionally, there has been the emergence of new attack types, such as Multi-stage Attacks (MSA), botnets, and SlowDoS attacks, which can mimic benign flow behaviors, rendering them difficult to detect, particularly when ML models were not specifically trained to recognize them [8], [9].

Accordingly, there is a possibility that new, unseen flows (including zero-day attacks) originate from a distribution distinct from what the ML models were trained on. Therefore, assessing the ML-IDS's ability to detect new, unseen flows, regardless of whether they have the same distribution as the training data or not, is a crucial metric commonly referred to as the ML model's ***generalization*** [10], [11]. To address the challenge of reduced generalization of detection performance in ML-IDS when dealing with diverse and unseen flows, it's essential to develop an ML model that can effectively discern abstract representations of flow statistical features.

Self-Supervised Learning (SSL) serves as an unsupervised learning approach that addresses the need for labeled data in supervised learning models by extracting meaningful representations from extensive unlabeled datasets. Additionally, SSL models enhance the generalization capability of unsupervised models by learning higher-level semantics and thereby obtaining more robust representations from the training data [12]. In SSL, for each sample, a label is automatically generated using its content. This label is utilized to minimize objective functions referred to as pretext tasks. These pretext tasks aim to learn a useful representation of data that can subsequently undergo fine-tuning for specific tasks, such as classification [13]. One of the well-known pretext tasks is contrastive learning, which aims to increase the mutual information between the representation of similar samples known as positive pairs [14]. Recently, contrastive methods have been used widely in computer vision and the natural language processing domain [13].

In this paper, we introduce a framework called *Self-Supervised Contrastive Learning-IDS (SSCL-IDS)*, which learns a generic representation of benign traffic. Figure 1 demonstrates the general architecture of the proposed *SSCL-IDS*. To extract the abstract representation of benign flows, we employ a technique where a subset of features from the anchor (original sample) is masked with their corresponding empirical marginal distribution to generate another view of the benign flow, termed its positive pair. Subsequently, leveraging a contrastive approach, the model is trained to minimize the distance between the anchor and its positive sample while maximizing the distance with other samples. The proposed *SSCL-IDS* enhances the ***generalization*** capabilities of the ML-IDS, thereby improving detection performance on ***unseen and sophisticated*** attacks, including MSA and slow-rate attacks. Moreover, it reduces the need for annotated network intrusion traffic. Furthermore, the knowledge obtained by *SSCL-IDS* can be transferred to a similar network intrusion classification task, particularly in scenarios where labeled network traffic samples are scarce. This technique, referred to as transfer learning, involves transferring learned representations from one model to another task. The model's adaptability is greatly enhanced by integrating transfer learning, enabling it to effectively detect emerging network intrusions and improve detection performance across diverse network traffic datasets.

In summary, the main contributions of this paper are as follows:

- We present *SSCL-IDS*, a model that exclusively utilizes benign flows and effectively addresses the scarcity of labeled attack flows. We apply two distinct corruption masks on both anchor and positive pairs to encourage the model to tackle a more challenging pretext task.
- We conduct a thorough evaluation of *SSCL-IDS*'s ability to generalize across diverse datasets, considering potential differences in their distributions.
- Demonstrating the efficacy of pretraining with *SSCL-IDS*, we exhibit results in transfer learning scenarios when dealing with a limited number of flow samples.

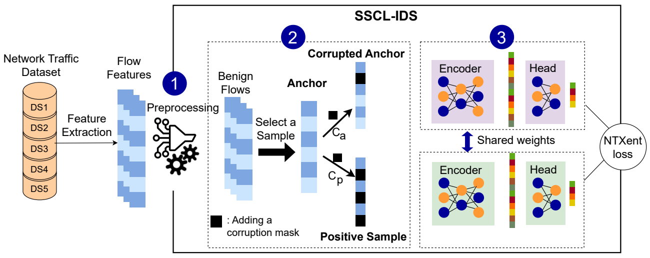

Fig. 1: The general architecture of the proposed *SSCL-IDS* with three main modules, including (1) preprocessing pipeline, (2) data augmentation, and (3) model training.

The paper is structured as follows: Section II introduces related works. Section III provides a brief overview of necessary background information. Section IV explains the architecture of the *SSCL-IDS*. In Section V, we conduct a comprehensive evaluation from different perspectives. Finally, in Section VI, we conclude this work.

II. Related Work

This section is divided into two categories of related works, including methods that improve the generalization using supervised ML models and ones that use SSL approaches.

A. Generalization Using Supervised ML-IDS

In [15], the authors designed a multi-aspect ensemble feature selection method to extract the most relevant features and improve the generalization of supervised ML-IDS. However, in their study, they could achieve good results from the datasets with similar attacks that have similar distributions. In [16], the optimization of hyper-parameters for supervised learning models was explored to enhance ML-IDS generalization. The study demonstrated the impact of dataset preprocessing and modifying the hyper-parameters on achieving higher detection performance. However, given the evaluations, which are limited to attacks from a single dataset, it is inferred that the improvements are targeted explicitly at intra-dataset generalization and detection performance. The authors in [10] explored the intra-dataset generalization of supervised ML-IDSs. They leveraged ensemble methods to improve generalization, constructing a pipeline comprising 12 supervised ML models from diverse categories.

The research results demonstrate that using the mentioned methods, it is possible to improve only the intra-dataset generalization of the supervised learning ML-IDS, which means that benign and attack samples are derived from either the same dataset or a dataset with a similar distribution.

B. Generalization Using Self-Supervise Learning

In [17], the authors used Graph Neural Networks (GNN) to capture flow features, including valuable network-related and edge information. They developed an SSL approach to improve ML-IDS detection generalization. Node auxiliary labels are extracted based on traffic volume, labeling nodes with high traffic as suspicious. However, this approach may only effectively detect volumetric attacks. Authors in [18] developed an ensemble feature selection approach to distinguish attack flows from benign ones utilizing anomaly identification, clustering, and classification. Furthermore, to learn the characteristics of benign and attack flows, they used a contrastive learning approach to maximize the distance between benign-attack pairs while minimizing the benign-benign and attack-attack pairs. To this aim, they trained their models using both benign and attack information. In [19], a supervised contrastive learning approach is conducted to perform the distance between features and labels in the shared embedding space. In this case, the sample features should be close to its ground-truth label (positive pair) while having a higher distance from the other labels (negative pairs). Therefore, ground-truth labels are required to create positive and negative pairs. Similarly, [20] employed a supervised contrastive learning approach to tackle imbalanced and constrained feature extraction capability issues. In this process, they used the dropout layer's randomness in the encoder for data augmentation, and the label information was utilized to establish positive/negative pairs. The authors in [21] aimed to overcome the limitation of lacking traffic labels by developing a self-supervised contrastive learning method. Their framework generated positive pairs by masking random packets within a flow. However, as indicated by [22], the data in the initial packets can have important statistical features; therefore, masking them could result in information loss. This approach was trained using both benign and attack flows. In this regard, authors in [5] introduced a self-supervised contrastive learning model in which the flow array was transformed into two-dimensional vectors, and various image-related augmentations, such as horizontal flip, vertical flip, random crop, and random shuffle, were applied to create positive/negative pairs. However, employing image augmentations on flow data points may generate semantically different pairs. Similar to [17], authors in [23] extracted features from each node using GNN. However, unlike [17], this work trained the SSL model in unsupervised settings. They used the random selection of the K neighbors of each node to create positive pairs. In this case, they masked information from some neighbors.

III. Background Concepts

This section briefly introduces SSL and transfer learning, along with the network traffic datasets used for training and evaluating the proposed *SSCL-IDS*.

A. Self-Supervised Learning (SSL)

In contrast to traditional unsupervised learning models that focus on learning the input data distribution, SSL models optimize pretext tasks by leveraging inherent properties and content of the data [13]. An example of pretext tasks includes recovering masked parts of a sample or assigning the same label to different augmentations of the sample [14]. Therefore, SSL has advantages, particularly when dealing with real-world data that lacks labeled annotations (e.g., network traffic). The SSL model can learn useful representations of the data, which can further be used for different tasks, such as transfer learning when few labels are available. The pretext task in the contrastive method is defined by maximizing a similarity score between similar samples (i.e., positive pairs) while minimizing it for samples with different semantics (i.e., negative pairs).

B. Transfer Learning

Transfer learning is a machine learning technique that utilizes knowledge from one domain (source) or task to another (target). It is mostly used when the target task has a limited amount of annotated data available [24]. Transfer learning has been extensively used in the computer vision and NLP domain, where weights of an ML model pretrained on the source dataset are used for prediction tasks on the target dataset [25]. Note that the pretraining can be both with or without label supervision from the source dataset. In this paper, we use contrastive SSL for the pretraining task.

C. Datasets

In this work, we utilize five distinct network traffic datasets to train and evaluate our proposed *SSCL-IDS*. The selection of datasets is based on the availability of their traffic data in PCAP format and the diversity of attacks they encompass (contains MSA, and slow-rate attacks). In the following, a brief explanation of each dataset is provided.

*1) **CICIDS2017 Dataset** [26]:* This dataset encompasses benign traffic captured by the abstract behavioral profiles of 25 users through protocols such as HTTP, HTTPS, FTP, SSH, and email. In addition to benign traffic, the dataset contains various types of attack traffic, including Web Attack, Infiltration, Botnet ARES, Brute Force, DoS, DDoS, SlowDoS, and Port Scan.

*2) **UNSW-NB Dataset** [27]:* This dataset consists of real benign traffic and synthetic attack behaviors collected in a controlled environment. The attacks are Fuzzers, Exploits, Worms, Shellcode, Backdoors, Reconnaissance, and DoS attacks.

*3) **CTU-13 Dataset** [28]:* This dataset consists of 13 scenarios of the botnet, benign, and background data. Benign traffic was extracted from university routers to represent real users' behavior. Each of the 13 botnet scenarios, Neris, Rbot, Virut, Menti, Sogou, Murlo, and NSIS.ay, is constructed to represent various malware behaviors.

*4) **CICDoS Dataset** [9]:* This dataset is designed to detect slow-rate DoS attacks. These attacks were combined with another dataset (ICSX 2012), which consists of high-rate DoS attacks. The attacks that are available in this dataset are Goldeneye, ddossim, hulk, Slowhttptest, RUDY, and Slowloris.

*5) **Botnet Dataset** [29]:* This dataset is designed to focus on generality, realism, and representativeness by incorporating diverse centralized and decentralized botnets utilizing various protocols. It comprises 16 botnets characterized by varying lifespans, accommodating both short and long-lived instances to enhance realism.

IV. SSCL-IDS METHODOLOGY

A. Problem Formulation

This section aims to explain the generalization challenge encountered in developing an ML-IDS. Initially, in Section III-C, we provided an overview of the datasets, characterized by various attack types and extracted from different network architectures, each of which can exhibit distinct behaviors, thereby indicating different distributions of functional flow features. To formalize this challenge, we define $D_s = \{X_s, P_s(x), x \in X_s\}$ as the source dataset and $D_t = \{X_t, P_t(x), x \in X_t\}$ as the target dataset. Due to potential differences in network types, service types, and data collection methods between the two datasets, their feature spaces (X_s and X_t) and probability distributions ($P_s(x)$ and $P_t(x)$) may vary. The generalization problem arises when an ML model f is trained on D_s using a learning algorithm A to minimize a loss function L, yet its performance may significantly degrade on D_t, resulting in $Performance(f, D_s) \neq Performance(f, D_t)$. This performance discrepancy signifies the presence of the generalization problem, wherein the model's efficacy on the training dataset fails to generalize well to a test dataset.

B. Proposed SSCL-IDS Framework

To improve the generalization of the ML-IDS and address the lack of labeled attack data, we propose the *SSCL-IDS*, which trains on only benign flows. As illustrated in Figure 1, the *SSCL-IDS* architecture comprises multiple modules, such as traffic preprocessing, data augmentation, and model training, which are explained further in the following.

*1) **Flow Scope & Preprocessing Pipeline**:* In the proposed *SSCL-IDS*, traffic flows are uniquely identified using their 5-tuple information, including source and destination IP addresses, source and destination port numbers, and the protocol type. To capture the statistical features of network flows, we utilize NFStream [30], a Python framework that can extract post-mortem and statistical flow features. The extracted features (88 features) are grouped into three categories: source to destination (src2dst), destination to source (dst2src), and bidirectional, encompassing packets moving in both src2dst and dst2src directions. Ground-truth labels are assigned based on information regarding victim hosts and attacker IP addresses provided by each dataset. To prevent information leakage, we implement a preprocessing pipeline across all datasets, which eliminates architecture-based features (5-tuple features) and all time-related features except for duration because these features could potentially reveal specific information about the network from which the traffic data originated. Additionally, features with a zero standard deviation are filtered out from the feature set because they cannot be informative. Following the preprocessing pipeline, the dataset comprises 45 features.

*2) **Data Augmentation - Corruption Mask**:* In SSL methods, augmentation generates additional training data by applying different transformations (here, we use a corruption mask) to the existing training data [13]. In *SSCL-IDS*, to create a positive pair (i.e., semantically similar samples), an augmented view of an anchor (original sample) is generated by incorporating the content of the sample. Each augmented sample represents a variant of the anchor sample with subtle differences, preserving essential semantic information. This approach ensures the creation of meaningful pairs for effective contrastive learning. In this work, to generate a positive pair, we randomly apply a corruption mask with two different corruption rates of C_p and C_a to two subsets P and A of the anchor's features. These subsets are randomly selected from the original set of features $F = \{f_1, f_2, ..., f_M\}$, where $|P| = C_p \times M$ and $|A| = C_a \times M$ respectively. The value of the j_{th} corrupted feature $\hat{f}_j$ is selected uniformly from the empirical marginal distribution of f_j; hence $\hat{f}_j \sim \text{Uniform}(f_j)$. The marginal distribution of each feature is initially calculated for the entire network traffic dataset. The method of extracting the value of the corrupted feature from the marginal distribution aligns with the structure of traffic flow features, which often includes diverse numerical scales and types. Integrating the positive corruption rate provides control over the dissimilarity among positive pairs. Larger values can alter all feature values, whereas smaller corruption rates only affect a limited subset of features. This results in a more straightforward optimization task and a less robust learned representation.

*3) **Model Architecture**:* In the proposed *SSCL-IDS* method, positive pairs created for a training sample are passed to an MLP-based encoder g with five hidden layers comprising 45, 64, 128, 64, and 45 neurons. The encoder's output is then fed through the head network h, including a projection head followed by a normalization layer, to prepare the features for calculating the contrastive loss (Figure 1). Note that for inference, the output of the encoder g is used for all the similarity calculations as well as fine-tuning in transfer learning tasks. Additionally, as shown in Figure 1, the encoder and projection head weights are shared to ensure the model learns consistent representations across different views of the same input.

*4) **Contrastive Loss Function**:* To encourage minimizing the distance between similar representations for positive pairs $(z_i, \hat{z}_i)$ while maximizing the distance between dissimilar representations for negative pairs (z_i, z_j), we use the NTXent (Normalized Temperature-scaled Cross-Entropy) loss function as formulated in equation 1 [13].

$$\text{NTXent}(z_i, \hat{z}_i) = -\log\left(\frac{\exp\left(\text{sim}(z_i, \hat{z}_i)/\tau\right)}{\sum_{j=1}^{2N} \exp\left(\text{sim}(z_i, z_j)/\tau\right)}\right) \quad (1)$$

where $(z_i, \hat{z}_i)$ refers to the representations of positive pairs while (z_i, z_j) refers to the presentations of the anchor and all the other samples in the mini-batch. Also, τ is the temperature parameter that scales the logits before applying the softmax activation function. It controls how close the similar data points should be. The smaller τ value leads to a higher penalty, thereby making the model place semantically similar data points closer to each other.

Algorithm 1 shows the proposed *SSCL-IDS*, including its corruption mask procedure, the output of encoder and head models, and how to incorporate these elements into the loss function.

Algorithm 1 SSCL-IDS Algorithm

```
**Input:** Training data $\mathcal{X} \subseteq \mathcal{R}^M$, Batch size $N$, temperature $\tau$, anchor corruption rate $C_a$, positive pair corruption rate $C_p$, encoder network $g$, head network $h$
$t_a = \lfloor C_a \times M \rfloor$
$t_p = \lfloor C_p \times M \rfloor$
$B = \{1, ..., M\}$
**for** sampled mini-batch $\{\mathbf{x}_i\}_{i=1}^{N}$ **do**
    **for all** $i \in \{1, ..., N\}$ **do**
        $A_i$: uniformly sample subset from $B$ of size $t_a$
        $P_i$: uniformly sample subset from $B$ of size $t_p$
        sample $j$ uniformly from $\{1, ..., M\}$
        **if** $j \notin A_i$ **then**    ▷ *Process of corrupting anchor*
            $\mathbf{a}_i^j = \mathbf{x}_i^j$    ▷ *Non-corrupted jth feature*
        **else**
            $\mathbf{a}_i^j = x_k^j$, where $x_k \sim \text{Uniform}(\mathcal{X})$
                ▷ *Corrupted jth feature*
        **end if**
        let $\mathbf{z}_{2i-1} = h(g(\mathbf{a}_i))$
        sample $j$ uniformly from $\{1, ..., M\}$
        **if** $j \notin P_i$ **then**    ▷ *Creating positive sample*
            $\mathbf{p}_i^j = \mathbf{x}_i^j$
        **else**
            $\mathbf{p}_i^j = x_k^j$, where $x_k \sim \text{Uniform}(\mathcal{X})$
        **end if**
        let $\mathbf{z}_{2i} = h(g(\mathbf{p}_i))$
    **end for**
    **for all** $i \in \{1, ..., 2N\}$ and $j \in \{1, ..., 2N\}$ **do**
        $s_{i,j} = \frac{\mathbf{z}_i^T \mathbf{z}_j}{(\|\mathbf{z}_i\| \|\mathbf{z}_j\|)}$
    **end for**
    let $l(i,j) = -\log \frac{e^{s_{i,j}/\tau}}{\sum_{k=1}^{2N} \nVdash_{k \neq i} e^{s_{i,k}/\tau}}$
    $L_{SSCL} = \frac{1}{2N} \sum_{k=1}^{N} \langle l(2k-1, 2k) + l(2k, 2k-1) \rangle$
**end for**
```

V. Evaluation Results

We evaluate the proposed *SSCL-IDS* on five different network traffic datasets explained in Section III-C. Additionally, we compare the generalization performance of the *SSCL-IDS* with two selected supervised and unsupervised baselines. Generalization performance refers to the model's detection performance on previously unseen samples from other network traffic datasets that may have different distributions. To perform the preprocessing and training, we utilize an Ubuntu server equipped with 250GB RAM and 4 GPUs (NVIDIA GeForce RTX 2080). The implementation is in Python, using the Scikit-learn, Pandas, and Pytorch libraries.

We assess *SSCL-IDS* detection performance with AUROC value, which is calculated across various thresholds (between 0 and 1), allowing us to evaluate the true positive rate against the false positive rate. Note that the positive class corresponds to the attack flows, while the negative class refers to the benign flows.

A. *Setup of Experiments*

To assess *SSCL-IDS* and facilitate comparative analysis, we briefly explain the designed baselines. Additionally, in this section, we explain the training and test datasets and the methodology for calculating the similarity metric in unsupervised learning models.

1) ***Baselines:*** To conduct a comparison, we choose a supervised ML model (*Multi-Layer Perceptron (MLP)*) and an unsupervised ML model (*AutoEncoder (AE)*) as baselines.

- **Unsupervised Setup:** An *AE* model, consisting of an encoder and a decoder, is known as an unsupervised learning model, focusing on optimizing data reconstruction. To classify the flows using *AE*, the distance of the new sample is computed from the embedded data, which is the output of the encoder. In this work, the designed *AE* model consists of a three-layer encoder including 64, 32, and 23 neurons and a three-layer decoder including 23, 32, and 45 neurons, which are selected using a cross-validation approach.
- **Supervised Setup:** A deep *MLP* model is selected as the supervised learning baseline. It consists of multiple layers to learn the connections between features and labels (Benign/Attack). To train this model, labeled data for both benign and attack classes is required. In this work, the designed *MLP* model comprises four layers with 64, 128, 64, and 32 neurons, respectively, which are selected using cross-validation approach.

2) ***Training Datasets:***

- **Training Dataset for *SSCL-IDS* & *AE*:** The training dataset used for unsupervised models of this work includes 60% of only benign flows from datasets.
- **Training Dataset for *MLP*:** Training supervised learning models require annotated data. Therefore, the training dataset for the chosen supervised model consists of 60% of the attack flows from CICIDS17, along with the training dataset used for the unsupervised models (*SSCL-IDS* and *AE*).

3) ***Test Dataset:*** The test dataset used for all evaluations includes 40% benign flows from each dataset that were not part of the training set. Furthermore, it involves all the attack traffic flows from all of the five datasets, which are entirely unseen for the *SSCL-IDS* and *AE* models.

4) ***SSCL-IDS Training Hyper-parameters:*** Following cross-validation on various hyper-parameters, a positive corruption rate of $C_p = 0.4$ is selected (the reason is discussed in Section V-C). The τ of the NTXent loss function in Equation 1 is set to 0.5. The embedding dimension remains the same as the original, i.e., $e_d = 45$. Moreover, the SSCL-IDS trains for 500 epochs on the batch size of $bs = 2046$.

5) ***Similarity Metric for Unsupervised Models:*** To evaluate the embedded data (as defined in Section V-A1) learned by *SSCL-IDS* and *AE*, we compute a similarity score $sim(.)$ by calculating the cosine similarity between the embedding vector g_{test} of test data x_{test} and embedding vector g_m of

TABLE I: Evaluating the detection performance of *SSCL-IDS* and comparing it with baseline models, including an unsupervised model (*AE*) and a supervised model (*MLP*).

Dataset	**AUROC Value (%)**		
	SSCL-IDS	Unsupervised Model	Supervised Model
CICIDS17	96.54	70.79	**97.25**
CICDoS	**87.73**	72.71	57.14
CTU-13	**99.85**	77.61	72.77
Botnet	**89.21**	73.48	60.64
UNSW-NB	**98.32**	65.64	60.59

all training data and finally take the maximum one. This calculation is performed using Equation 2.

$$sim(x_{test}) = \max_m\left(\frac{g_{test}^T g_m}{\|g_{test}\|\|g_m\|}\right) \tag{2}$$

According to Equation 2, it is possible to compute the directional similarity between the embedded benign training data and the embedded benign or attack data in the unseen test dataset. The expectation is that attack samples will exhibit a higher cosine similarity score, indicating that they are oriented in a different direction compared to benign samples.

B. Detection Performance of SSCL-IDS & Baselines

To evaluate the detection performance of the proposed *SSCL-IDS* and baselines, including *AE* (unsupervised) and *MLP* (supervised) models, the AUROC value is computed (as defined in Section V). Table I demonstrates the detection performance of these models across all five datasets. According to the results, the proposed *SSCL-IDS* demonstrated better detection and generalization performance across most datasets, except for the CICIDS17 dataset, where it achieves the second-best result, closely trailing the supervised model (*MLP*). It's worth noting that the MLP was trained on labeled data, where the model was exposed to the attack samples of the CICIDS17 dataset (mentioned in Section V-A2). However, training on labeled data may not entirely reflect real-world conditions. The results of the supervised model's detection performance on the other datasets show its generalization performance, which is the lowest, as expected. On the other hand, the proposed *SSCL-IDS* model achieved up to 27% better generalization performance compared to the supervised model (among all datasets). Furthermore, the *SSCL-IDS* outperforms the unsupervised model across all datasets by up to 15%. We argue that the better generalization of *SSCL-IDS* stems from differences in its learning processes. In *SSCL-IDS*, contrastive learning encourages the model to push together the representation of the samples with similar semantics in the embedding space proven to learn more effective signals of training data semantics [12], [13]. This approach enables the model to learn a more effective representation of benign flows in the training data.

C. Impact of the Corruption Rate on Detection Performance

As explained in Section IV, the anchor corruption rate (C_a) and positive pair corruption rate (C_p) are fundamental in *SSCL-IDS* architecture. While C_p controls the similarity of constructed positive pairs, C_a introduces additional noise to the original data, challenging the *SSCL-IDS* model, leading to an improvement in its detection performance generalization. Through our experiments, we observed that selecting C_a greater than 0.2 reduces the model's detection performance due to excessive noise added to the original data. Therefore, we set $C_a = 0.2$ for all evaluation results. Additionally, varying the value of C_p influences the corruption of a larger subset of features, generating dissimilar positive pairs and impacting the optimization process. A higher C_p makes the optimization task more challenging, while a smaller C_p results in a less robust representation. Figure 2 illustrates the impact of different C_p values on the final detection performance, measured through AUROC values, with the experiment conducted three times independently. According to the findings presented in Figure 2, a C_p value of 0.4 consistently yields the highest detection performance across all datasets. In contrast, a very small C_p value, specifically $C_p = 0.2$, results in a lower AUROC value than $C_p = 0.3$ and $C_p = 0.4$. Therefore, we choose $C_p = 0.4$ as the optimal corruption rate for the final evaluation results of *SSCL-IDS*.

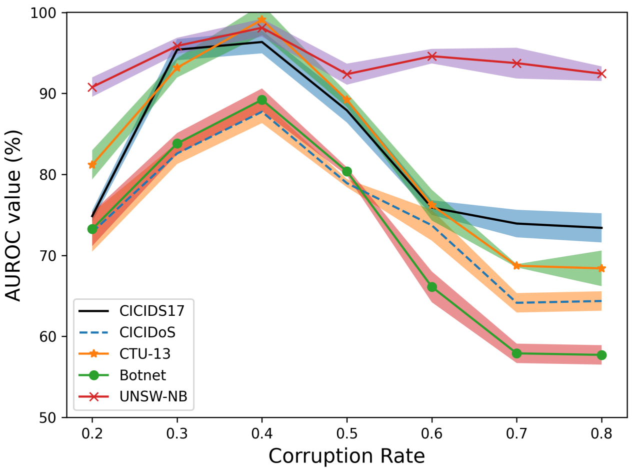

Fig. 2: Impact of positive pair corruption rate (C_p) on detection performance. The experiment is carried out over three independent runs, and the figure depicts the mean value and standard deviation for each dataset.

D. t-Distributed Stochastic Neighbor Embedding (t-SNE)

In this experiment, we utilize the t-SNE, a dimensionality reduction technique, to depict the embedding data generated by the proposed *SSCL-IDS* in a lower-dimensional space [31]. Furthermore, it can show the quality of the learned embeddings, indicating how effectively the model has captured the underlying structure of the data. When similar examples are clustered together in the t-SNE plot, it can be understood that the proposed *SSCL-IDS* has successfully captured relevant features and relationships within the data. Figure 3 illustrates the t-SNE plot for both the raw data and the embedded data (output of the *SSCL-IDS*) within the CICIDS17 dataset. As shown, the data points of each class (Benign/Attack) in the plot with embedded data are closely grouped, with a more clear separation between classes and fewer overlaps compared to when t-SNE is applied on

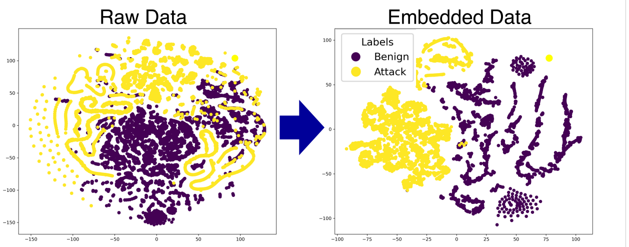

Fig. 3: Comparing the embedded data from *SSCL-IDS* with the raw data of the CICIDS17 dataset in a lower-dimensional space. The output of *SSCL-IDS* (the right one) illustrates a clear distinction between attack and benign data points.

raw data. This separation enhances the ability to distinguish between attack and benign flows, contributing to improved model detection performance.

E. Analysing the Generalization of Detection Performance

In the prior experiments (Sections V-B and V-C), we assessed the generalization of *SSCL-IDS* detection performance using 40% of unseen benign flows and all attack data from each dataset. In this section, we expand our evaluation to a more challenging scenario where the models have been trained with benign flows of **only one** of the datasets and evaluated with all datasets. The outcomes of this scenario are depicted in Figure 4, with the upper figure representing *SSCL-IDS* results and the lower figure depicting the supervised model results. Comparing these two figures reveals an improvement in the generalization of the *SSCL-IDS* compared to the supervised learning model. Hence, it demonstrates that *SSCL-IDS* effectively learns the characteristics of benign flows, leading to enhanced generalization in detection performance. Nevertheless, as expected, the diagonal values, which represent the AUROC values of a test set split from the same dataset as the training set, are lower in *SSCL-IDS* compared to the supervised learning model. This arises because *SSCL-IDS* exclusively trains on the benign flows in the training set, while the supervised learning model trains on both benign and attack flows. Moreover, as illustrated in Figure 6, we extend the evaluation to analyze how adding different numbers of datasets (which only contain benign samples) during training affects the detection performance of *SSCL-IDS*. The number of datasets increases following the sequence outlined in Table I. To clarify, the initial dataset (One DS) is *CICIDS17*, with *CICDOS* added in the second dataset (Two DS), and so on. As shown in Figure 6, the AUROC value for each data set increases when it is included in the corresponding training dataset. Notably, *CICDOS* shows an increase in AUROC value when included in the Two DS training, and a similar improvement is observed for *CTU13* in the Three DS training with the addition of its dataset. Likewise, the detection performance of *UNSW-NB* shows a high value when the *SSCL-IDS* is trained with all datasets, including its training dataset. We argue that adding more datasets with benign flows encourages the model to learn a more generic understanding of the benign traffic, thus enabling the use of transfer learning methods.

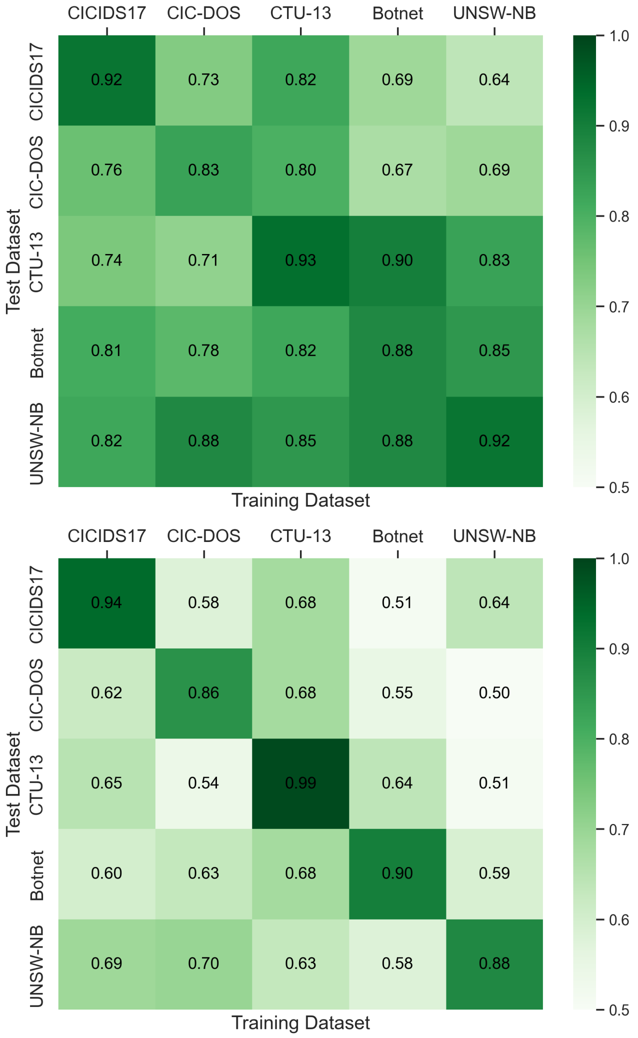

Fig. 4: Comparing the generalization performance of *SSCL-IDS* (upper figure) with the *supervised* ML-IDS (lower figure). Both ML models are trained on one dataset and tested on others.

F. Sample Efficiency of SSCL-IDS for Transfer Learning

Transfer learning, as defined in Section III-B, is a machine learning technique that uses the knowledge of a pretrained model to perform a new task in the case of labeled data scarcity. In this section, our focus is on sample efficiency, particularly examining the number of labeled samples needed for transfer learning when employing *SSCL-IDS* as a pretrained model. To demonstrate the effectiveness of transfer learning, we train an MLP model (with the same configuration as outlined in Section V-A1) twice: once on raw data (RD) and another time on the embeddings of the training data (ED) generated by the pretrained *SSCL-IDS*.

To assess the sample efficiency for each dataset, labeled samples are added incrementally to the training dataset starting from the 10^{-5} portion of the original dataset size, which consists of only 7 to 20 labeled samples (depending on the dataset's size). Figure 5 illustrates a comparison of the detection performance using the pretrained *SSCL-IDS* (embedded data) and when the model is trained on the

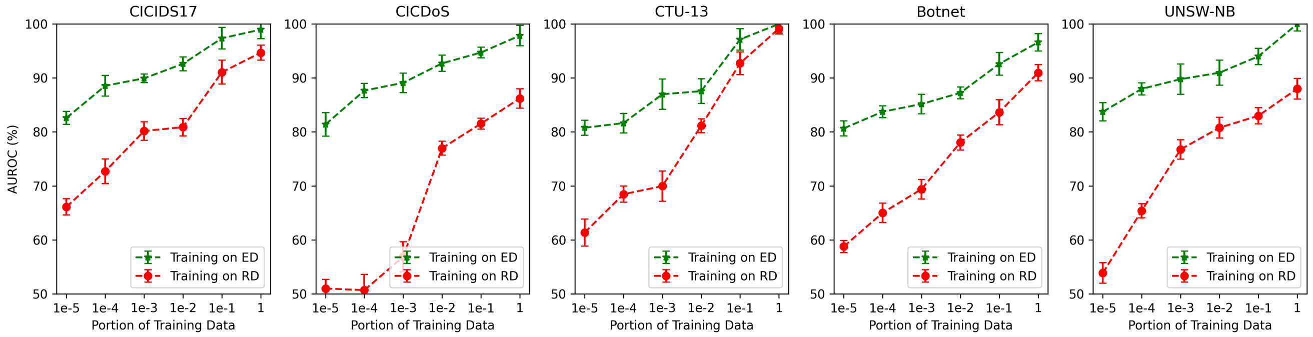

Fig. 5: Comparing label efficiency for the benign/attack classification task, 'ED' denotes embedded data, and training on it shows the detection performance when transfer learning is employed. 'RD' refers to raw data, and training on it reflects supervised ML-IDS results.

raw data. As depicted in Figure 5, across all datasets, the AUROC values show higher detection performance when the MLP model is trained on the pretrained *SSCL-IDS*. The gap is bigger when only a few labeled samples are used for training. For instance, the detection performance for two datasets *CICDoS* and *UNSW-NB* when training on less than 20 labeled samples of raw data is close to random guessing (AUROC=50%) while using pretrained embeddings results in AUROC values above 80%. Additionally, it is noteworthy that higher detection performance is attained when utilizing all labeled samples for training with pretrained *SSCL-IDS* compared to training with raw data.

G. Comparing SSCL-IDS with SSL-based Related Works

In this section, we compare our proposed *SSCL-IDS* with other SSL-based related work. This comparison is made using their reported values. Consequently, variations in preprocessing pipelines among different works may affect their final results. Table II presents a comparison of supervised SSL (first category) and unsupervised SSL (second category) methods utilizing two evaluation metrics, AUROC and F1-score, on all mentioned datasets. Although some related works evaluated their models on the NSL-KDD dataset, we could not do so due to the unavailability of its network traffic in PCAP format. In fact, to enhance the flexibility of *SSCL-IDS* for evaluating generalization performance on new datasets, we implemented a preprocessing pipeline (detailed in Section IV-B1) that initiates receiving traffic files. Furthermore, based on the available information, there is no SSL-based approach that has been evaluated on the *CICDoS* and *CTU-13* datasets. According to Table II, some of the supervised SSL methods that used label information (for generating positive/negative pairs) in the data augmentation process achieve higher detection performance than *SSCL-IDS*, which doesn't use label information for creating positive pairs. Note that *SSCL-IDS* outperforms the majority of supervised SSL methods on the *UNSW-NB* dataset. However, *SSCL-IDS* outperforms all the other unsupervised SSL-based related works, except for *BYOL* on the *Botnet* dataset. Moreover, we evaluate our method on a broader range of datasets compared to existing approaches, showcasing its generalization in detection performance across varied scenarios. For instance, the *CTU-13* dataset contains botnets that are part of multi-stage attacks (MSA), known for their challenging detection characteristics. In addition, the *CICDoS* dataset includes SlowDoS attacks, which further demonstrates the effectiveness of our approach in detecting sophisticated attacks.

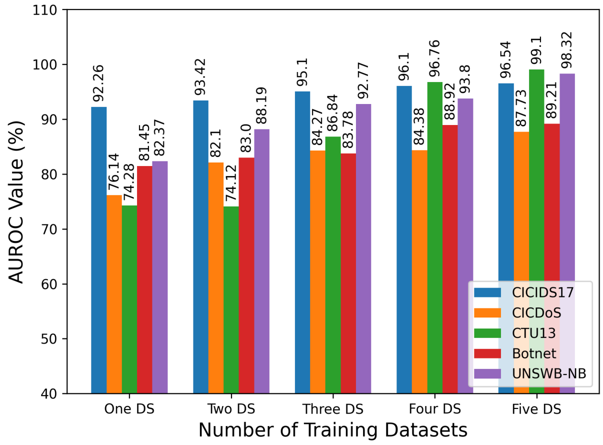

Fig. 6: Comparing AUROC values for each dataset when the number of training datasets increases. In the X-axis of the figure, 'DS' refers to the dataset.

TABLE II: Comparison of *SSCL-IDS* with related works.

Related Work	**AUROC / F1-Score (%)**				
	CICIDS17	CICDoS	CTU-13	Botnet	UNSW-NB
Supervised SSL:					
RLB-CL [19]	- / 90.72	-	-	- / 85.65	- / 89.42
ConFlow [20]	- / 99.96	-	-	- / 99.16	-
CLDNN [21]	- / 99.96	-	-	- / 99.47	- / 92.91
TS-IDS [17]	98.80 / 99.55	-	-	97.14 / 96.67	99.86 / 99.75
Unsupervised SSL:					
BYOL [5]	96.0 / 95.48	-	-	97.0 / 98.46	88.0 / 92.41
Anomal-E [23]	- / 90.72	-	-	-	- / 92.18
SSCL-IDS (ours)	96.54 / 97.73	87.73 / 95.47	99.85 / 98.32	89.21 / 97.16	98.32 / 99.43

VI. CONCLUSION

In our study, we highlight the challenges encountered by Machine Learning-based Intrusion Detection Systems (ML-IDS) for detecting network-based anomalies in practice. For instance, coping with heterogeneous diverse network traffic distribution patterns, accurately detecting attacks that mimic benign flows, and the scarcity of labeled attack traffic to build effective ML-IDS. Such obstacles often hinder the ML-IDS's ability to generalize effectively and achieve robust detection performance on unseen and zero-day attacks due to their absence in the training data. To address these issues, we introduced *SSCL-IDS*, a novel approach that adapts self-supervised contrastive learning techniques to boost the generalization capabilities of ML-IDS. Our proposed method is trained exclusively on benign traffic, mitigating the challenges posed by the scarcity of labeled and unknown/zero-day attacks. The comprehensive results demonstrate the detection and generalization improvement of *SSCL-IDS* compared to both supervised (which leverage labels) and unsupervised baseline methods. Additionally, by transferring the representations learned by *SSCL-IDS* to a new network traffic dataset utilizing only a few of its training samples (< 20), we achieve AUROC scores exceeding 80% where without fine-tuning the average AUROC score on different datasets is close to random guess. This highlights the adaptability of *SSCL-IDS* in detecting emerging network intrusions and previously unseen traffic data. While *SSCL-IDS* exhibits strong detection performance, contrastive learning lacks control over negative samples. Thus, integrating a strategy to select negative samples can enhance the model's learned representation.

Acknowledgment

This work is funded by the Federal Ministry of Education and Research of Germany (BMBF) through the CELTIC-NEXT Flagship Project AI-NET-PROTECT and Software Campus Grant 01IS17050 (ML-based NIDS).

References

[1] "Cisco Annual Internet Report (2018–2023) ," (Accessed on 13.11.2023). [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] Y. Guo, "A review of machine learning-based zero-day attack detection: Challenges and future directions," *Computer Communications*, vol. 198, pp. 175–185, 2023.

[3] P. Panagiotou, N. Mengidis, T. Tsikrika, S. Vrochidis, and I. Kompatsiaris, "Host-based intrusion detection using signature-based and ai-driven anomaly detection methods," *Information & Security*, vol. 50, no. 1, pp. 37–48, 2021.

[4] M. Hajizadeh, S. Barua, and P. Golchin, "Fsa-ids: A flow-based self-active intrusion detection system," in *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2023, pp. 1–9.

[5] Z. Wang, Z. Li, J. Wang, and D. Li, "Network intrusion detection model based on improved byol self-supervised learning," *Security and Communication Networks*, vol. 2021, pp. 1–23, 2021.

[6] M. Almseidin, J. Al-Sawwa, M. Alkasassbeh, and M. Alweshah, "On detecting distributed denial of service attacks using fuzzy inference system," *Cluster Computing*, vol. 26, no. 2, pp. 1337–1351, 2023.

[7] P. Golchin, J. Weil, R. Kundel, and R. Steinmetz, "Dynamic network intrusion detection system in software-defined networking," in *2nd Workshop on Machine Learning & Networking (MaLeNe), co-located with the 5th International Conference on Networked Systems (NetSys 2023)*, 2023.

[8] M. Lefoane, I. Ghafir, S. Kabir, and I.-U. Awan, "Multi-stage attack detection: Emerging challenges for wireless networks," in *2022 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2022, pp. 01–05.

[9] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting http-based application layer dos attacks on web servers in the presence of sampling," *Computer Networks*, vol. 121, pp. 25–36, 2017.

[10] L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Inter-dataset generalization strength of supervised machine learning methods for intrusion detection," *Journal of Information Security and Applications*, vol. 54, p. 102564, 2020.

[11] S. Layeghy, M. Baktashmotlagh, and M. Portmann, "Di-nids: Domain invariant network intrusion detection system," *Knowledge-Based Systems*, vol. 273, p. 110626, 2023.

[12] N. Rafiee, R. Gholamipoor, N. Adaloglou, S. Jaxy, J. Ramakers, and M. Kollmann, "Self-supervised anomaly detection by self-distillation and negative sampling," in *International Conference on Artificial Neural Networks*. Springer, 2022, pp. 459–470.

[13] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "Simclr: A simple framework for contrastive learning of visual representations," in *International Conference on Learning Representations*, vol. 2, 2020.

[14] D. Bahri, H. Jiang, Y. Tay, and D. Metzler, "Scarf: Self-supervised contrastive learning using random feature corruption," *arXiv preprint arXiv:2106.15147*, 2021.

[15] P. Golchin, R. Kundel, T. Steuer, R. Hark, and R. Steinmetz, "Improving ddos attack detection leveraging a multi-aspect ensemble feature selection," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2022, pp. 1–5.

[16] A. Sarkar, H. S. Sharma, and M. M. Singh, "A supervised machine learning-based solution for efficient network intrusion detection using ensemble learning based on hyperparameter optimization," *International Journal of Information Technology*, vol. 15, no. 1, pp. 423–434, 2023.

[17] H. Nguyen and R. Kashef, "Ts-ids: Traffic-aware self-supervised learning for iot network intrusion detection," *Knowledge-Based Systems*, vol. 279, p. 110966, 2023.

[18] Q. Liu, D. Wang, Y. Jia, S. Luo, and C. Wang, "A multi-task based deep learning approach for intrusion detection," *Knowledge-Based Systems*, vol. 238, p. 107852, 2022.

[19] M. Lopez-Martin, A. Sanchez-Esguevillas, J. I. Arribas, and B. Carro, "Supervised contrastive learning over prototype-label embeddings for network intrusion detection," *Information Fusion*, vol. 79, pp. 200–228, 2022.

[20] L. Liu, P. Wang, J. Ruan, and J. Lin, "Conflow: contrast network flow improving class-imbalanced learning in network intrusion detection," *Research Square Preprint*, 2022.

[21] Y. Yue, X. Chen, Z. Han, X. Zeng, and Y. Zhu, "Contrastive learning enhanced intrusion detection," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4232–4247, 2022.

[22] P. Golchin, C. Zhou, P. Agnihotri, M. Hajizadeh, R. Kundel, and R. Steinmetz, "Cml-ids: Enhancing intrusion detection in sdn through collaborative machine learning," in *2023 19th International Conference on Network and Service Management (CNSM)*. IEEE, 2023, pp. 1–9.

[23] E. Caville, W. W. Lo, S. Layeghy, and M. Portmann, "Anomal-e: A self-supervised network intrusion detection system based on graph neural networks," *Knowledge-Based Systems*, vol. 258, p. 110030, 2022.

[24] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[25] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," *Journal of Big Data*, vol. 9, no. 1, p. 102, 2022.

[26] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

[27] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[28] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *computers & security*, vol. 45, pp. 100–123, 2014.

[29] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *2014 IEEE Conference on Communications and Network Security*. IEEE, 2014, pp. 247–255.

[30] Z. Aouini and A. Pekar, "Nfstream: A flexible network data analysis framework," *Computer Networks*, vol. 204, p. 108719, 2022.

[31] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html