IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 14 December 2023; revised 6 April 2024; accepted 13 May 2024.
Date of publication 16 May 2024; date of current version 22 May 2024.

The associate editor coordinating the review of this article and approving it for publication was S. Scott-Hayward.

Digital Object Identifier 10.1109/TMLCN.2024.3402158

Improving Generalization of ML-Based IDS With Lifecycle-Based Dataset, Auto-Learning Features, and Deep Learning

DIDIK SUDYANA[1], YING-DAR LIN[1] (Fellow, IEEE), MIEL VERKERKEN[2], REN-HUNG HWANG[3] (Senior Member, IEEE), YUAN-CHENG LAI[4], LAURENS D'HOOGE[2], TIM WAUTERS[2] (Member, IEEE), BRUNO VOLCKAERT[2] (Senior Member, IEEE), AND FILIP DE TURCK[2] (Fellow, IEEE)

[1]Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan
[2]IDLab-imec, Department of Information Technology, Ghent University, 9000 Ghent, Belgium
[3]College of Artificial Intelligence, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan
[4]Department of Information Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan
CORRESPONDING AUTHOR: D. SUDYANA (dsudyana@cs.nycu.edu.tw)

ABSTRACT During the past 10 years, researchers have extensively explored the use of machine learning (ML) in enhancing network intrusion detection systems (IDS). While many studies focused on improving accuracy of ML-based IDS, true effectiveness lies in robust generalization: the ability to classify unseen data accurately. Many existing models train and test on the same dataset, failing to represent the real unseen scenarios. Others who train and test using different datasets often struggle to generalize effectively. This study emphasizes the improvement of generalization through a novel composite approach involving the use of a lifecycle-based dataset (characterizing the attack as sequences of techniques), automatic feature learning (auto-learning), and a CNN-based deep learning model. The established model is tested on five public datasets to assess its generalization performance. The proposed approach demonstrates outstanding generalization performance, achieving an average F1 score of 0.85 and a recall of 0.94. This significantly outperforms the 0.56 and 0.42 averages recall achieved by attack-based datasets using CIC-IDS-2017 and CIC-IDS-2018 as training data, respectively. Furthermore, auto-learning features boost the F1 score by 0.2 compared to traditional statistical features. Overall, the efforts have resulted in significant advancements in model generalization, offering a more robust strategy for addressing intrusion detection challenges.

INDEX TERMS Intrusion detection, ML-based IDS, model generalization, lifecycle-based dataset, auto-learning features.

I. INTRODUCTION

INTRUSION Detection Systems (IDS) play a crucial role in safeguarding network systems. Broadly speaking, it can be categorized into either signature-based or anomaly-based. Signature-based IDS, although effective at identifying known threats, often fail to detect novel attacks. On the other hand, anomaly-based IDS can detect new types of attacks, but they suffer from a high false-negative rate [1].

Machine Learning (ML) has emerged as a solution to these problems, significantly improving the performance of IDS. It boosts accuracy, lowers the false-negative rate, and enables the detection of a wider range of threat variants [2]. This is achieved by allowing the ML model to learn the behavior of the network, which it uses to classify and identify potential threats.

Research in the field of ML-based IDS has proposed numerous solutions, with the current research focus being on improving the accuracy of the ML-based IDS [3], [4], [5], [6]. However, achieving high accuracy is only a part of the solution. A robust IDS should also exhibit strong generalization capabilities [7], [8]. Generalization refers to the model's ability to apply knowledge from training data to

unseen data. Unfortunately, many of the current studies have trained and tested models only on a single dataset that did not truly represent unseen data, creating a trustworthiness issue. The ability to accurately classify new data using a different kind of datasets beyond the training data is crucial for the ML-based IDS. Without testing on the genuinely unseen data, the actual performance of these systems remains unclear, leaving the true effectiveness of the IDS systems in real-world scenarios ambiguous.

The current generalization test utilizes the inter-dataset strategy by training ML models with one dataset and testing them with another, such as training on dataset A and testing on dataset B. However, existing research using this approach shows strong performance on trained data but poor results on unseen data [9], [10], [11]. For example, models trained with CIC-IDS-2017 and tested with CIC-IDS-2018, and vice-versa, achieved the F1 score below 50%.

This decrease in performance is alarming, especially considering the CIC-IDS-2017 and CIC-IDS-2018 datasets are very similar, containing the same categories of attacks and using identical tools. The primary difference between them lies in the scale of their respective network environments, which ranges from a configuration of 15 machines to one comprising 500 machines. Despite these close similarities, the observed decline in generalization capability poses a significant challenge in the domain of ML-based IDS. These results imply that even minor changes in the network environment, such as differences in scale or traffic volume, can drastically affect a model's ability to effectively adapt to new situations.

This raises serious concerns about the reliability of ML-based IDS in real-world applications. If these systems are unable to maintain consistent performance in slightly varied network settings, their usefulness in diverse and changing real-world scenarios becomes questionable. The primary goal of ML-based IDS is to reliably identify threats across different and evolving environments, maintaining a steadfast level of security regardless of changes in network infrastructure. Therefore, addressing this issue of poor generalization is vital for the development of robust and reliable IDS solutions.

The use of synthetically generated datasets in prior studies, despite not fully replicating real network benign traffic, is a critical initial step in validating models. Ensuring a model's performance in a controlled environment is essential for establishing its foundational robustness before deployment in more complex, real-world scenarios. However, evidence from the literature suggests that achieving broad generalization, even in these controlled settings, remains challenging.

The poor generalization in IDS is influenced by several factors, including the nature of the dataset, feature extraction methods, and machine learning model choices. Many previous studies used attack-based datasets, categorizing attacks only by their type [12], leading to datasets that are overly specific to a particular attack tool and lack diversity [12]. Current feature extraction methods rely either on statistical approaches that capture spatial information but miss other crucial details, or on packet-level features, which represent packet data within a traffic flow using an array of header fields. Moreover, many IDSs employ shallow learning models which tend to underperform compared to deep learning counterparts [5], [13], [14].

Our research is specifically motivated by the observed gaps in the generalization performance of these DL models. The prevalent practice of training and testing models on the same dataset does not adequately challenge their ability to generalize to truly unseen data, a critical aspect for real-world applications. This work aims to rigorously evaluate ML/DL models' generalization capabilities by employing inter-dataset testing, an area where existing literature indicates a notable shortfall. Our investigation extends beyond statistical features to include raw packet data, aiming to fortify the argument for this research's necessity in advancing IDS efficacy against novel threats.

While previous research predominantly focused on using attack-based datasets, statistical features, and shallow machine learning (ML) models or deep learning (DL) models, which had a poor generalization performance, this study introduces a novel composite approach in the dataset, features, and learning model. First, the CREMEv2 dataset [15] is used as the training data. This dataset introduces the lifecycle-based dataset method, which emphasizes both attack lifecycles and techniques. It maps attack lifecycles to specific tactics and techniques, based on the MITRE-ATT&CK framework. Tactics represent the 'why' behind an attack, signifying its purpose. They encompass objectives an attacker aims to achieve, such as initial access, execution, persistence, and impact. Techniques, on the other hand, focus on the 'how'–the specific methods employed to execute a given tactic. By detailing the process of an attack based on its lifecycle, the dataset achieves greater variety, making it richer and more comprehensive.

The approach also incorporates auto-learning features to extract features directly from raw traffic without heavily depending on predefined statistical methods. Additionally, a deep learning CNN model is used to create traffic patterns from these features, enhancing its ability to detect anomalous network activities [16], [17].

The effectiveness of the proposed approaches is evaluated by thoroughly testing them using the inter-dataset strategy, employing five of the latest public datasets for a comprehensive assessment. This rigorous testing ensures a thorough understanding of how well the system generalizes to unseen data.

To enhance the understanding, this work also investigates: (1) factors influencing generalization: datasets, features, and learning models, (2) impact of dataset: attack-based vs. lifecycle-based datasets. The model is trained with CIC-IDS-2017, CIC-IDS-2018 as the attack-based datasets, and CREMEv2 as the lifecycle-based dataset to grasp how dataset choice affects generalization, (3) features and learning models, assessing the performance of different methods for generalization, as well as key auto-learning features in lifecycle-based datasets, revealing how CNN effectively

interprets data, (4) exploring how the number of convolution layers affects model generalization.

With relation to the related works, this paper's contributions are three-fold:

- To the best of our knowledge, this is the first study that proposes a novel composite approach by integrating a lifecycle-based dataset, auto-learning features, and CNN to improve the generalization performance of ML-based IDS.
- A detailed evaluation and analysis of generalization performance provides insights and validation for the proposed methods.
- Sensitivity analysis of deep learning model configurations is performed to analyze their impact on generalization performance.

The contribution of this paper lies not only in analyzing the benefits of utilizing the MITRE-ATT&CK framework-based dataset for improving ML-based IDSs but also in demonstrating a methodological advancement in model training and evaluation. Integrating auto-learning features within a CNN addresses the challenges of model generalization across previously unseen datasets. This method significantly enhances model generalization, reduces reliance on extensive feature engineering, and illustrates how IDS can evolve from depending on manually defined, static features to dynamically extracting and learning features directly from raw data. This work not only advances cybersecurity by providing a nuanced approach to enhancing IDS capabilities but also equips researchers and practitioners with robust tools to tackle real-world variability in security threats.

The remaining sections of this paper are structured as follows: Section II discusses related work, providing an overview of previous research in the field. Section III delves into the system architecture, presenting the overall framework used in this study. In Section IV, the system implementation is detailed, explaining the methodology employed. Section V is devoted to presenting the results and analysis derived from this experiments. Finally, in Section VI, conclusions are drawn based on the findings, highlighting the key insights and contributions of this research.

II. RELATED WORKS

Table 1 provides a comprehensive summary of prior research on model generalization for ML-based Intrusion Detection Systems (IDS). This summarization offers a comparison based on several crucial parameters, including the generalization testing/training method employed, features utilized, the ML/DL model chosen, datasets considered, and the solutions proposed. In current literature, there are four different methods for evaluating model generalization performance: intra-dataset testing, unified-dataset testing, inter-attack testing, and inter-dataset testing.

In the intra-dataset testing, both training and testing are conducted on the same dataset. Meanwhile, the unified-dataset testing combines several datasets and proceeds to train and test on this unified dataset. The inter-attack testing strategy involves training exclusively on a single attack class within a dataset and subsequently testing against other attack classes within that same dataset. Lastly, inter-dataset testing diverges from the previous methods as it entails training and testing on entirely separate datasets. This method is particularly crucial for assessing the adaptability and robustness of the ML-based IDS across different data sources.

For the intra-dataset testing, various studies have been conducted. Reference [19] focused on exploring pattern leakage across three distinct datasets. Pattern leakage, in this context, refers to the process of encoding and scaling the entire dataset. Remarkably, this process also encompassed the test set derived from the full dataset. By testing with statistical features and conventional ML models, they were able to achieve high accuracy. Meanwhile, in [18], the emphasis was on combining ML + DL methods, which showed promise in increasing generalization. This approach resembled that of [20], which aimed to improve generalization by selecting appropriate statistical features. However, similar to [19], the testing process in [18] and [20] was also unreliable since the model was trained and tested using the same dataset, resulting in the test dataset not being truly unseen.

In the context of unified-dataset testing, current literature presents two approaches for combining datasets: Federated Learning (FL) [21], [22], and feature engineering [23]. FL involves training local models from different datasets and aggregating them into a global model using the FedAdagrad method. This approach enhances generalization capabilities by integrating knowledge from diverse datasets. On the other hand, feature engineering aims to unify features and integrate datasets for training and testing. While training and testing with combined data show potential, this method proves impractical for real-life applications due to the extensive dataset preparation required, making the process time-consuming and resource-intensive.

For the inter-attack testing, [24] utilized the integrated feature selection techniques, specifically filtering and embedding methods. They employed the CIC-DDoS-2019 dataset for training with the LGBM model on one DDoS class and subsequently tested the model on other DDoS classes within the same dataset. However, a limitation of this approach is that it is only applicable to similar types of attacks, such as DDOS-like attacks, and may not be suitable for assessing generalization across significantly diverse attack types.

In the context of inter-dataset testing, [9] and [10] investigated generalization using supervised learning. They trained the model with CIC-IDS-2017 and tested it with CIC-IDS-2018, and vice versa. However, the model exhibited poor generalization on the testing data in both cases. Similar results were observed in [11], where unsupervised learning was utilized for testing, but the model still failed to generalize to unseen data. The F1 score of these inter-dataset testing works remained below 60%.

In contrast, this paper differs from previous works in several ways. While previous studies relied on attack-based dataset methods, which directly execute attacks, resulting in limited effectiveness for handling unseen data, this work employs the CREMEv2 dataset as the lifecycle-based dataset,

TABLE 1. Summary of the related works on generalization of ML-Based IDS.

Paper	Generalization Testing Method	Features	Model	Dataset			Solution
				Training	Testing	Dataset Type	
[18]	Intra-dataset testing	A	DL	MAWILab-2018, ISCX-2012		Attack-based	Combine CVAE and RF
[19]		S		NSL-KDD, UNSW-NB15, KDD-99			Investigate pattern leakage
[20]				MalMem2022, CIC-IDS-2017, CIC-DDoS-2019, UNSW-NB15			Involve causal and noise features
[21]	Unified-dataset testing		FL	BoT-IoT, TON-IoT, UNSW-NB15, CIC-IDS-2018			Involve cross-silos FL strategy
[22]				ToN-IoT, CIC-IDS-2018, BoT-IoT (only DDoS)			Involve sharing information during FL training process
[23]				UGR-16, UNSW-NB15, NSL-KDD			Use feature engineering to integrate 3 datasets
[24]	Inter-attack testing		ML	CIC-DDoS-2019			Train and test with different DDoS attacks
[9]	Inter-dataset testing			CIC-IDS-2017	CIC-IDS-2018		Investigate the generalization effect
[10]				CIC-IDS-2018	CIC-IDS-2017, CIC-DDoS-2019		Investigate the effect of data-constrained on generalization
[11]				CIC-IDS-2017	CIC-IDS-2018		Investigate unsupervised learning on generalization
Ours		A	DL	CREMEv2	5 public datasets	Lifecycle-based	Involve lifecycle-based dataset, auto-learning features, and CNN

A/S: Auto-learning, Statistical DL/FL/ML: Deep Learning, Federated Learning, Machine Learning

extracts features using an auto-learning technique, and the CNN autonomously learns the features and data. The model's generalization performance is further validated through inter-dataset testing.

III. SYSTEM ARCHITECTURE

This section provides an overview of the three fundamental elements employed to enhance generalization performance. Firstly, an explanation of CREMEv2 as a lifecycle-based dataset method is presented. Next, the discussion covers the auto-learning features and the CNN architecture used in this study.

A. LIFECYCLE-BASED DATASET: CREMEv2

Lifecycle-based datasets such as CREMEv2 are structured to encompass the full spectrum of a cyber threat, mapping out each stage of an attack from inception through execution and aftermath, following the structure of the MITRE ATT&CK framework. This framework provides a globally accessible knowledge base of adversary tactics and techniques, which CREMEv2 utilizes to generate realistic sequences of cyber attack lifecycles. Instead of focusing solely on the impact of an attack, these datasets offer insights into the 'behavior' of threat actors over time, illustrating the progression and evolution of attacks across different stages. By integrating data that spans the entire lifecycle of an attack, these datasets enable the development of IDS capable of identifying early indicators of a threat, understanding the sequential tactics of attackers, and adapting to the changing dynamics of cyber threats. This comprehensive view, guided by the structured phases of the MITRE ATT&CK framework, allows for the creation of security systems that are not only reactive but also proactive, significantly enhancing their ability to detect and mitigate a broad spectrum of cyber threats, especially those that are new or in the process of evolution.

In contrast, attack-based datasets are designed to capture specific instances of cyber threats, concentrating primarily on the technical details of those threats, such as signatures, methods, and immediate impacts. These datasets are constructed by simulating particular types of attacks, which focus on the impact of the attack itself, resulting in narrow data. The primary aim is to provide a repository of attack signatures that can be used to train IDS to recognize specific types of malicious activities. While this approach is effective for detecting well-known, predefined attack patterns, it inherently lacks the broader context in which actual cyber-attacks unfold. Consequently, IDS developed using attack-based datasets may perform well in identifying specific threats for which they have been trained but can struggle with new or varied attack forms that deviate from the documented patterns.

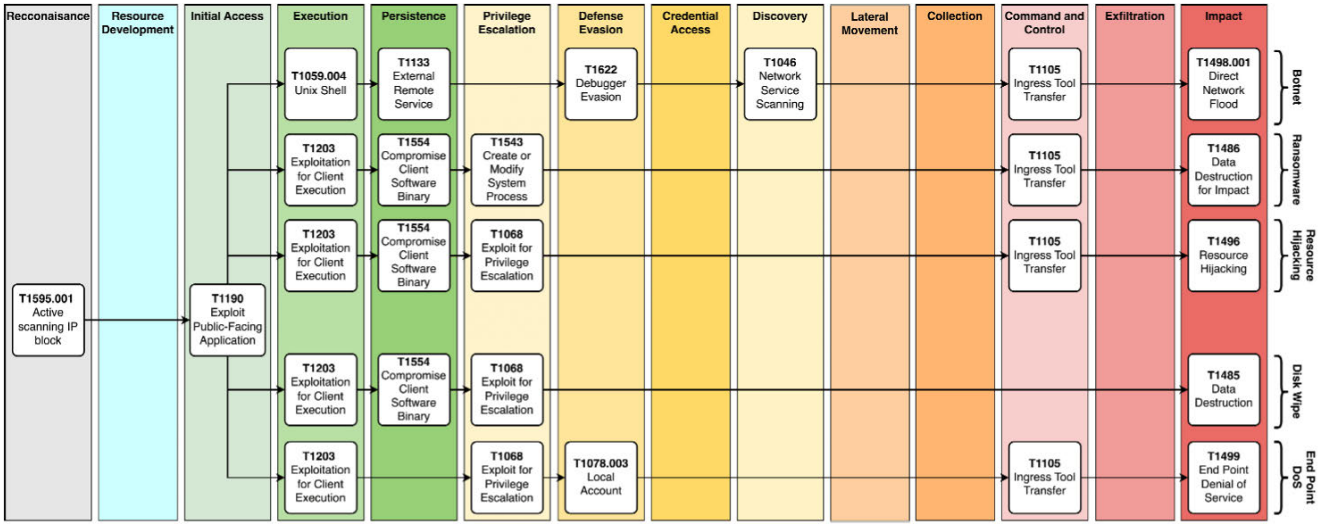


FIGURE 1. **Mapping attacks to sequences of techniques on CREMEv2 [15].**

The fundamental difference between the two types of datasets lies in their scope and applicability: while attack-based datasets offer depth in specific attack types, lifecycle-based datasets provide a broad and integrative perspective that encompasses the full range of cyber threat behaviors. This comprehensive approach inherent in lifecycle-based datasets results in better-prepared, more adaptable cyber defense mechanisms capable of addressing the multifaceted nature of modern cyber threats.

CREMEv2 was generated based on the MITRE ATT&CK framework [15]. This framework serves as the foundation for developing specific threat models and methodologies.

CREMEv2 comprises five attack variants, encompassing both host-based and network-based attacks. These variants include ransomware, resource hijacking, mirai, disk wipe, and end point dos. To map these attack variants to 14 tactics within 17 different MITRE techniques, a manual comparison was conducted based on attack behavior. The mapping results are shown in Figure 1, where the columns represent the used tactics, and the rows correspond to the techniques or lifecycles involved.

As an updated version of CREMEv1 [25], CREMEv2 involves a broader range of techniques with a restructured testbed [15]. This new version incorporates a router machine and establishes a host-only network to segregate the main OS from the virtual machines, thereby mitigating external attack impacts. The testbed comprises ten virtual machines, including one controller, one data logger, four clients, and one machine simulating the roles of attacker, target, and benign server. This setup aims to create a controlled environment conducive to reproducing various cyber-attack scenarios while efficiently monitoring and logging all pertinent data.

For the replication of attack scenarios, CREMEv2 utilizes several attack tools aligned with the MITRE ATT&CK framework. These tools simulate various types of cyber threats, such as botnets, disk wipes, ransomware, resource hijacking, and endpoint DoS attacks. For example, Mirai's pre-compiled version is employed for botnet simulations, Metasploit modules for gaining privileged access, and custom scripts for simulating ransomware and resource hijacking activities. Each tool is chosen based on its ability to represent a specific attack technique within the framework, ensuring accurate mapping of these activities to MITRE ATT&CK techniques.

The dataset generation process begins with the automated replication of attack scenarios using the aforementioned tools. The attack replication is orchestrated by the controller server, which manages all processes and controls all entities involved. As the attacks unfold, data is concurrently recorded by the client, target server, and benign server. These data streams are then transmitted to the data logger server for centralized log collection. Utilizing a centralized log collection system facilitates efficient data processing and enables the generation of comprehensive datasets that accurately reflect the intricacies of each attack scenario. Following data collection, a meticulous labeling process is undertaken using the breakpoint information generated during the attacks.

Figure 2 illustrates the generation of lifecycles and the collection of traffic in CREMEv2. It details the interactions between various components, such as attacker and benign servers, malicious clients, target server, data logger, and controller. These components simulate cyber-attacks following specific stages indicated by the legend, ranging from reconnaissance to executing impact tactics like data destruction. The legend explains the sequence and types of attacks, highlighting tools and methods used at each stage.

B. AUTO-LEARNING FEATURES

In this work, 'auto-learning' is referred to as the system's capability to autonomously learn and extract relevant features from raw data, specifically network traffic, without the need for pre-defined rules or manual feature selection. This is accomplished through the utilization of a CNN, inherently designed to recognize patterns and features

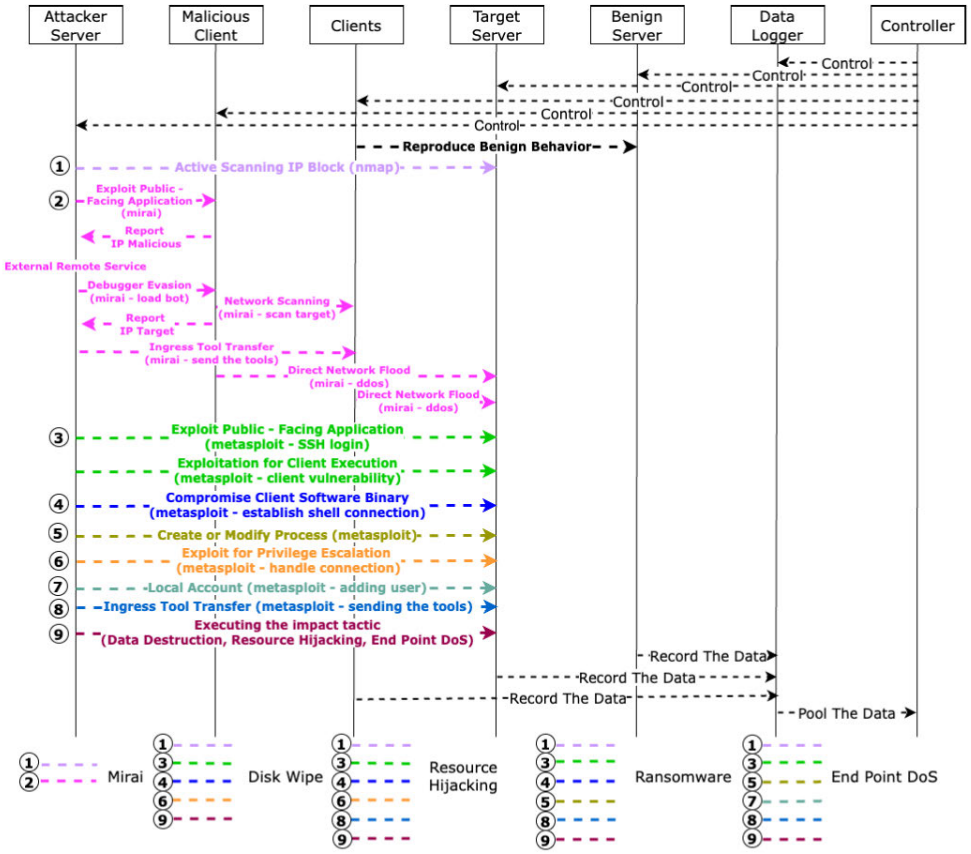


FIGURE 2. **CREMEv2 lifecycles reproduction workflow.**

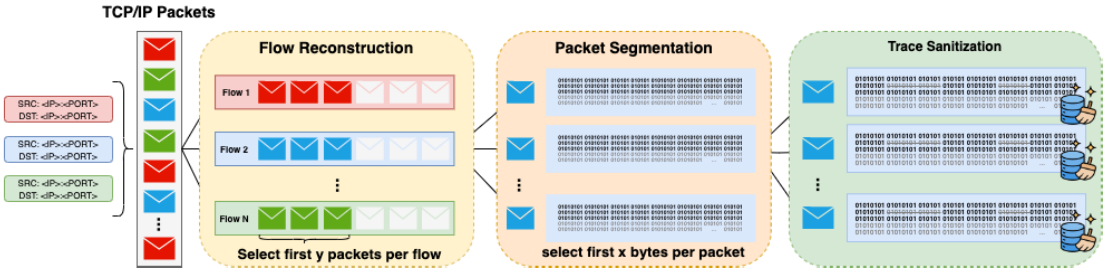


FIGURE 3. **Auto-learning feature extraction process.**

directly from the input data during its training. The technique's 'learning' aspect is characterized by the CNN model's ability to autonomously identify patterns, anomalies, and characteristics within the network traffic's raw byte representations, facilitated by transforming the raw traffic data into an image-like format. This enables the convolutional layers to detect features across multiple levels of abstraction.

The approach to handling raw network traffic data aims to transform it into a format efficiently processed by the CNN model, involving the sampling of data within a network flow into a predetermined number of packets and bytes [16]. In networking, a 'flow' refers to a sequence of packets between a specific source and destination, typically identified by shared network attributes such as IP addresses, port numbers, and protocol types. The strategy of extracting only the initial x bytes from the first y packets of each flow aims to reduce computational and memory demands, thus ensuring more efficient processing.

This approach strikes a balance between capturing enough detail to identify anomalies and avoiding information overload. The initial bytes can provide a snapshot of the traffic, offering enough insight for anomaly detection without the need for full packet inspection.

Figure 3 provides an illustration of the feature extraction auto-learning process. This process begins with the flow reconstruction for identifying the flow from the raw data, achieved by grouping raw network packets based on their 5-tuple characteristics. The first y packets per flow are subsequently extracted.

The next phase involves packet segmentation, where the first x bytes of the initial packet from each flow are extracted. This extraction process serves two purposes: (1) it reduces the traffic load for analysis and expedites the process, particularly for lengthy sessions, (2) because deep learning models require uniform data lengths, a default packet size is established. Packets surpassing this predefined size are trimmed, while smaller ones are zero-padded. This technique involves the extraction of the initial three packets, each consisting of 60 bytes, as it has been empirically determined to yield the best performance in terms of F1 score, as indicated by prior research conducted by [16].

The final step in the feature extraction process involves trace sanitization. This process includes removing all specific testbed configuration and metadata information by deleting 24 bytes of such information from each packet in the input data. This step is vital to ensure that the CNN model does not resort to shortcut learning based on artifacts in the training data, such as ports, MAC addresses, and IP addresses. The sanitization ensures that the CNN model does not acquire knowledge of specific testbed configurations. Following this sanitization, the data is then converted into one-dimensional vectors, which are then employed as inputs for the CNN model.

Based on this auto-learning configuration, which extracted three packets, each consisting of 60 bytes and removed certain metadata information, a feature set of 107 features is obtained, with each feature representing information from the bytes of individual packets. Features with index 0-35 represent the first packet, 36-71 correspond to the second packet, and 72-107 pertain to the third packet.

C. CNN ARCHITECTURE

In the realm of IDS, it has been observed that unsupervised methods, despite their advantages, struggle with generalization, particularly during inter-dataset testing, where a noticeable drop in performance was observed [11]. Consequently, a supervised learning approach has been adopted, characterized by using diverse datasets, advanced feature extraction techniques, and CNN models to improve generalization. This decision is driven by the necessity for precise, actionable insights in cybersecurity, emphasizing the accurate identification of normal and malicious activities. By capitalizing on the structured framework of supervised learning, a more effective approach to tackling the challenges of intrusion detection has been established.

The CNN model is one of the prominent deep learning models known for its exceptional performance in utilizing auto-learning features. Through this approach, the model autonomously learns valuable features, showcasing its effectiveness in various applications.

In this study, a one dimensional (1D) CNN model was utilized. The effectiveness of 1D-CNNs in handling temporal relations can be attributed to their ability to capture local dependencies and identify discriminative patterns within sequences. Unlike 2D-CNNs, which process two-dimensional spatial data, 1D-CNNs are designed to process one-dimensional sequential data, making them particularly suitable for tasks involving time series or sequential inputs. In the context of network traffic analysis, 1D-CNNs can effectively capture spatial dependencies between adjacent bytes in network packets. This capability allows them to discern patterns that are indicative of different classes of protocols or applications, thereby enabling accurate classification of traffic [26].

TABLE 2. CNN architecture.

Layer	Type	Filters/neurons	Stride	Padding
1	single-ConV + ReLu + Batch Normalization	32 (kernel size=6)	1	Same
2	Maxpooling + Batch Normalization	Kernel size=2	2	Same
3	Flatten + Batch Normalization	-	-	-
4	Dense + Batch Normalization	1024	-	-
5	Dense + Batch Normalization	10	-	-
6	Dense	1	-	-

The architecture of the CNN model is presented in Table 2, showcasing its key components and design. The values and configurations chosen for each component in this table have been carefully selected based on extensive experimentation, ensuring optimal performance and generalization capabilities. A single convolutional layer is utilized to effectively extract and learn essential features from the input data. Additionally, batch normalization is implemented after each crucial layer, a strategy that not only mitigates overfitting but also addresses the vanishing gradient problem, thus enhancing the model's generalization capability. The integration of max pooling further contributes to the architecture's efficiency by summarizing the learned features and reducing computational complexity.

The architecture also includes a flattening process followed by three dense layers designed for learning. Rectified Linear Unit (ReLU) is employed as the activation function, contributing to the model's learning process. In the final dense layer, a sigmoid activation function is used, facilitating binary classification of the data. This architecture embodies a comprehensive approach to extracting, processing, and classifying data, ensuring optimal performance for intrusion detection tasks.

IV. SYSTEM IMPLEMENTATION

To conduct the experiments, several steps are followed. Initially, the testing datasets are selected, and the re-labeling process is applied to all the testing datasets. The relevant features are then extracted. While extracting the features, both auto-learning features and statistical features are utilized. The inclusion of statistical features aims to facilitate a performance comparison between auto-learning and statistical features. Afterward, the data extracted from the auto-learning features is trained using a CNN model, and its generalization performance is assessed. Furthermore, we train models with statistical feature data using both CNN and ML models. This decision is motivated by the fact that

many previous studies have utilized standard ML models to evaluate their generalization performance [9]. Thus, to ensure a fair comparison of results, it is included in the evaluation process. Finally, the obtained results are analyzed.

In this section, the details of the various components used in the experiments are explained, including the selection of testing datasets, the re-labeling process of those datasets, information on auto-learning and statistical features, and the specific ML model utilized for training the statistical features.

A. SELECTING TESTING DATASET

For assessing the generalization performance, it is crucial to incorporate a variety of testing datasets to ensure a comprehensive evaluation. In this study, the selection of testing datasets was guided by specific criteria. Public datasets available in recent years were considered, prioritizing those that offered raw pcap files and comprehensive documentation detailing data collection procedures. This documentation encompassed insights into the tools, techniques, data capture environment, and timestamps of attack processes. Given that auto-learning features require raw data processing and re-labeling, access to this documentation was crucial. Based on these criteria, five datasets were identified that met the requirements. The following section provides a concise overview of the testing datasets utilized in this study:

- **CIC-IDS-2017**: Released in 2017 by the Canadian Institute for Cybersecurity (CIC) [27], this dataset had 7 distinct attack classes. It was generated over the course of 5 days, utilizing a network of 14 machines. Aside from offering a comprehensive list of attack classes, it provides raw PCAP files and has been extensively utilized by numerous studies.
- **CIC-IDS-2018**: As the extension of the CIC-IDS-2017, this dataset was published in 2018, encompassing the same attack classes. However, it differentiates itself by magnifying the testbed using a cloud environment, thus involving over 500 machines in a span of a 10-day attack process.
- **CIC-DDOS-2019**: Published in 2019 by the CIC [28], this dataset has a specialized focus on DDoS variants. It comprises 12 different DDoS variants, making it an invaluable resource for specialized DDoS research.
- **CREMEv1**: Published in 2022, CREMEv1 is a sophisticated toolchain designed for automatic dataset collection [25]. The dataset contains multiple data sources, including network traffic, host statistics, and Syslog with 5 attack scenarios. Each of these scenarios is executed in a systematic manner, spanning across three predefined stages.
- **CCU Mirai HTTP**: This dataset published in 2019 by CCU, Taiwan [16]. It was specifically designed to emulate DDoS attacks and was set up using the Mirai malware by involving 7 IoT devices. These devices were used to initiate 4 types of DDoS attacks, with a comprehensive list detailing each DDoS variant.

These datasets provide a comprehensive and diverse testing ground to assess the model's resilience and adaptability to varying attack profiles and complexities, ensuring a robust evaluation of its generalization capabilities.

B. DATASET RE-LABELING PROCESS

Re-labeling the raw data necessitates several steps. Initially, the process began with the raw pcap files, with the goal of extracting features from their raw data and then performing the re-labeling process. However, because these pcap files mixed benign and attack data, labeling posed a challenge. To streamline this process, it was necessary to filter and separate benign and attack traffic, group them based on each class, and save each class to a new file.

To facilitate this separation, crucial details such as attack duration, attacker and victim IP addresses were derived from the dataset metadata. Utilizing this information, the separation process was initiated according to the methodology outlined in [29]. For instance, for the CIC-IDS-2017 dataset regarding the DoS-slowloris attack, it was determined that the specific attack time was from July 5th, 9:47 to 10:10 a.m., with the attacker IP being 205.174.165.73 and the victim IP being 192.168.10.50. These details were utilized to group relevant packets from the original pcap files. By extracting these packets and re-writing them into new pcap files, the data was effectively organized and labeled. This re-labeling process step was crucial in creating structured datasets for subsequent analysis and model training. It also guarantees the dataset's integrity and reliability for subsequent analysis.

C. FEATURE EXTRACTORS

In the process of extracting features, two different feature extraction methods were utilized: auto-learning features as the default configuration and statistical features for performance comparisons. The details of the implementation of these feature extractors are explained below.

AUTO-LEARNING FEATURES

In the processing of auto-learning features, Scapy [30] was used to handle the raw data stored in PCAP files. Scapy is a versatile packet manipulation tool that offers several advantages in the data processing realm. Using Scapy greatly facilitated the workflow by allowing us to read the raw data, carry out flow reconstruction, select the first three packets per flow, extract the initial 60 bytes from each packet, and subsequently sanitize the data.

STATISTICAL FEATURES

Two different statistical feature extraction tools were utilized. The purpose was to compare their performance against the auto-learning features. The tools utilized were CICFlowMeter and NFStream, both of which are widely recognized within the field.

CICFlowMeter, developed by CIC, generates a comprehensive set of 81 features [31]. It has been integrated into feature extraction processes across all CIC datasets. This tool is particularly notable for its user-friendly interface and widespread adoption by researchers. The fixed version

of CICFlowMeter was utilized, as derived from [29], ensuring consistency and reliability in the extraction process.

Another feature extraction tool is NFStream, which was officially released in 2022 [32]. Operating on a Python-based platform, NFStream is capable of calculating the statistical attributes of network flows. It offers a total of 61 features and boasts efficient data structures and algorithms that enable it to handle high-speed network traffic effectively, both in online and offline scenarios.

D. ML ALGORITHM

As previously noted, besides training the lifecycles-based dataset with CNN for auto-learning and statistical features, machine learning (ML) was also used to train the statistical features. In this study, XGBoost was specifically selected for the training of statistical features due to its notable benefits over other ML algorithms [33], [34]. XGBoost's strength lies in its architecture, where it combines multiple weak learners to form a more robust predictive model. This construction enables it to effectively handle complex relationships within the data.

When evaluating the model, F1 score was chosen as the primary metric due to its ability to strike a balance between precision and recall. By combining both precision and recall into a single score, the F1 score provides a comprehensive assessment of a model's ability to correctly classify instances across different classes.

E. MODEL TRAINING AND TESTING PIPELINES

Figure 4 outlines the pipeline of the model development process, dividing it into two distinct pipelines: one leveraging statistical features and the other leveraging auto-learning features. In the first pipeline, raw data in pcap files undergo feature extraction using NFStream and CICFlowMeter tools, which extract statistical features for model training and evaluation. The CREMEv2 dataset is then split into three parts: 70% for training, 20% for validation, and 10% for testing with a random split method. The training data is utilized to train the XGBoost and CNN models, while the validation data serves to fine-tune model hyperparameters and optimize performance. Finally, other datasets are utilized for inter-dataset testing to assess the models' robustness and scalability across different datasets.

The second pipeline employs auto-learning features, transforming the raw PCAP files into a suitable format for the CNN model. Similar to the previous pipeline, the CREMEv2 dataset is randomly split into 70% for training, 20% for validation, and 10% for testing. The training data is utilized to train the CNN model, while the validation data is employed to fine-tune model hyperparameters and optimize performance. Additionally, other datasets are utilized for inter-dataset testing. The end goal is to assess and compare the generalization performance of ML and CNN models, highlighting the potential of auto-learning features in improving the generalization of the IDS model. All the code can be retrieved from the GitHub repository [35].

V. RESULTS AND ANALYSIS

In this section, the results are structured around the factors impacting the generalization performance. These include (1) the most important factor in improving generalization performance, encompassing datasets, features, and learning models; (2) the influence of datasets - attack-based datasets such as CIC-IDS-2017 and CIC-IDS-2018 versus the lifecycle-based dataset, CREMEv2 - on generalization performance; (3) an evaluation of features and learning models, specifically, the performance of auto-learning and statistical features trained using lifecycle-based datasets with ML or CNN models, along with an examination of key features of auto-learning with CNN in CREMEv2 to understand how CNN interprets and classifies unseen data; and (4) an analysis of parameter effects, focusing on the impact of convolution layer configurations and the number of packets and bytes of auto-learning features on the model's generalization capabilities.

A. DATASETS Vs. FEATURES Vs. LEARNING MODELS

To evaluate the impact of different factors on enhancing generalization performance, a comprehensive analysis was conducted, by comparing combinations of datasets, features, and learning models. Figure 5 illustrates the results of the generalization testing performance for all these combinations, with the average F1 score being derived from five different testing datasets. It is important to note that auto-learning is dependent on the CNN model, which results in six performance combinations.

The results clearly demonstrate that the choice of dataset as the training data is the most crucial factor influencing the improvement of generalization, followed by the features and type of learning models. The combination of a lifecycle-based dataset with auto-learning features and CNN demonstrated a remarkable average F1 score of 0.85, effectively generalizing across all testing datasets. However, a significant degradation in performance was observed when transitioning from the lifecycle-based dataset to the attack-based training dataset (CIC-IDS-2017), dropping the F1 score from 0.85 to 0.45. Even in the worst-case scenario with the lifecycle-based dataset using statistical features (0.69 and 0.67), it still outperformed the best configuration in the attack-based dataset (0.45).

Datasets serve as the foundational knowledge base for machine learning models. The creation of a dataset with appropriate methods equips the model with a thorough understanding of various scenarios. On the other hand, features, which are derived representations of the raw data to highlight certain aspects or patterns in the data, are constrained by the quality and range of the original dataset. Superior feature extraction methods on a limited dataset, such as utilizing attack-based dataset with auto-learning features and CNN, will still fall short compared to even basic features from a comprehensive dataset such as the lifecycle-based dataset with statistical features. Furthermore, when considering learning models, different models vary in their ability to discern patterns. However, even the most

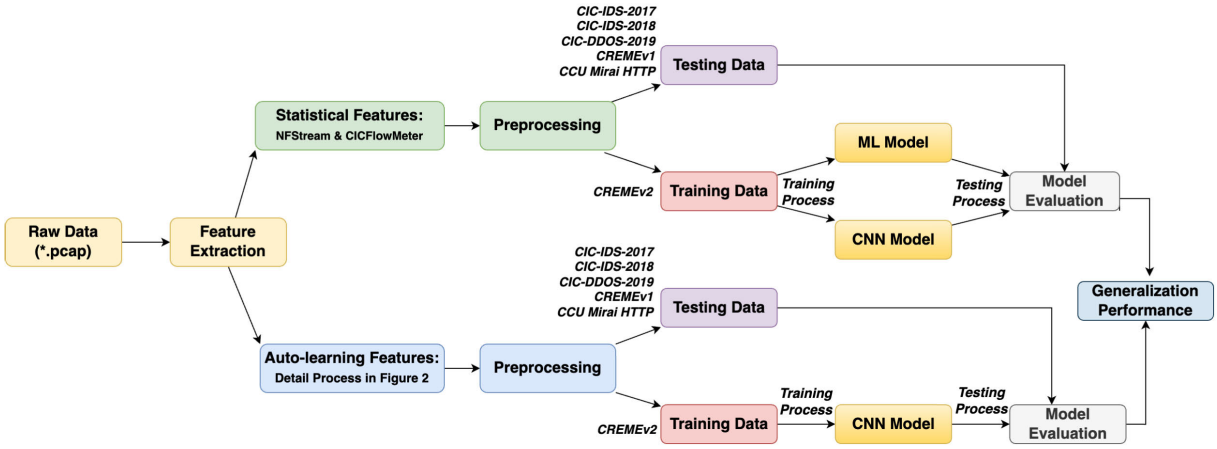


FIGURE 4. **The pipelines of training and testing the IDS model.**

sophisticated model, when trained on a limited dataset, can only learn from what it is exposed to. Its ability to generalize well is thus inherently limited by the dataset's scope and quality.

Training a model on the CREMEv2 as a lifecycle-based dataset not only empowers it to identify the detailed behaviors and sequences that characterize attack lifecycles but also enriches its understanding of malicious activities beyond the limitations of specific attack scenarios. This broad perspective enables the model to recognize potential threats within unfamiliar attack-based data, applying the core principles learned from CREMEv2 effectively, despite having no prior exposure to their specific signatures or instances. Furthermore, incorporating auto-learning features into the CNN model significantly bolsters this capability. By enabling the model to abstract and learn from input data representations through multiple layers, it can detect a wide range of features, from low-level details to high-level features. This comprehensive approach to data interpretation greatly enhances the model's accuracy and its ability to adapt to emerging and diverse cyber threats, showcasing a remarkable improvement in its generalization capabilities.

In conclusion, while features and learning models play significant roles in the generalization performance, a dataset with a lifecycle-based approach stands out as the most crucial element. Ensuring dataset quality and representativeness by considering an attack as part of its lifecycle should thus be a primary concern in ML-based IDS research aiming for model generalization. In subsection V-B, the reasons behind the superior generalization capabilities of the lifecycle-based dataset for the attack data are explored. Meanwhile, a thorough analysis of the features is provided in subsection V-C.

B. DATASET: ATTACK-BASED Vs. LIFECYCLE-BASED

In this analysis, the comparative performance between using attack-based and lifecycle-based datasets is explored. For this purpose, CIC-IDS-2017 and CIC-IDS-2018 were used as examples of attack-based datasets, while CREMEv2 served

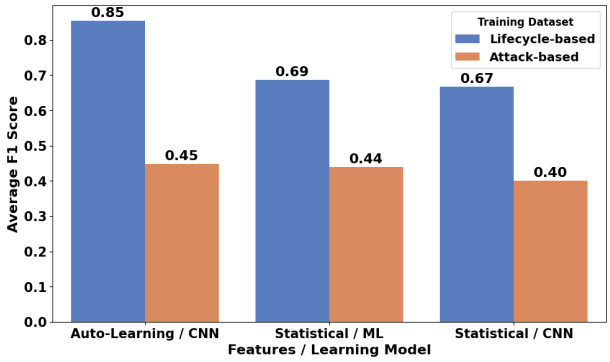


FIGURE 5. **Datasets vs. Features vs. Learning Models.**

as the representative for the lifecycle-based dataset. Detailed testing results and insights can be found below, with Figure 6 providing an overview of the generalization testing outcomes for each dataset.

Training the model with either CIC-IDS-2017 or CIC-IDS-2018 as attack-based datasets, employing CNN with auto-learning features, results in a failure to effectively generalize over the majority of the testing datasets. These models exhibit exceptional performance in intra-dataset tests, surpassing 0.99. However, their performance significantly degrades when subjected to inter-dataset testing. Specifically, training with CIC-IDS-2017 leads to an average F1 score of 0.44 in inter-dataset tests, with an inability to recognize CREMEv2 dataset (0.109). Similarly, training with CIC-IDS-2018 results in an average F1 score of 0.421 in inter-dataset tests, indicating a failure to generalize over these datasets.

In contrast, training the model with the lifecycle-based dataset, CREMEv2, resulted in the best generalization performance, achieving an average F1 score of 0.85 in inter-dataset testing. This approach excelled in recognizing attack data within the testing dataset, as indicated by its high recall score, averaging at 0.94 (as depicted in Figure 5b). This substantial improvement becomes evident when compared to attack-based datasets, which only achieved average recall scores of 0.56 and 0.42 when trained with CIC-IDS-2017

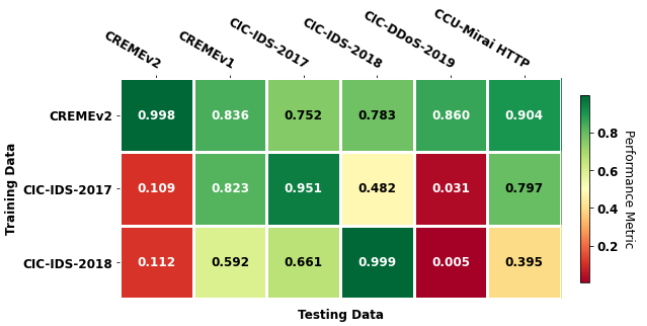
and CIC-IDS-2018, respectively. The increased recall rate highlights the superior ability of the model to detect attacks, underlining its potential for enhancing intrusion detection systems. This performance was two times better than using an attack-based dataset for training, and notably outperformed previous works [9], [10], [11], whose inter-dataset testing recall score hovered around 0.4-0.6.

The strength of the lifecycle-based dataset lies in its comprehensive characterization of attacks into techniques and lifecycles, providing a more detailed view of attack behaviors. This approach enables a deeper understanding of attack patterns and behaviors, facilitating the identification of both known and new or similar attack behaviors. The richness and variety of attack behavior within the lifecycle-based dataset contribute significantly to its heightened generalization power.

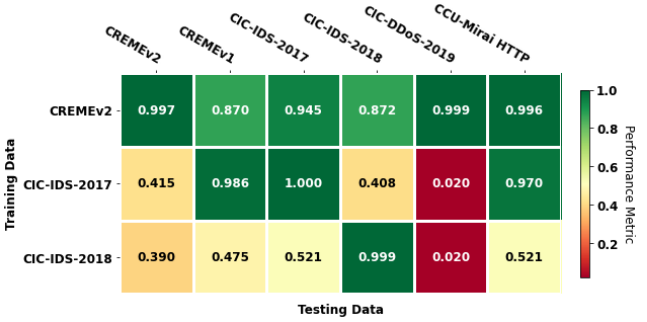
Furthermore, the patterns observed in the sequences of techniques within CREMEv2 align closely with those found in other attack datasets, facilitating the model's ability to generalize across diverse attack scenarios. For instance, in the case of SSH brute-force attacks from the CIC-IDS datasets, CREMEv2 captures similar behaviors through techniques such as "Exploit Public Facing Application," wherein unauthorized access to SSH servers is attempted. Similarly, the port scan attacks in CIC-IDS find correspondence with the "Active Scanning" technique in CREMEv2. Although CREMEv2 may not contain attacks labeled with the exact names as those in other datasets, the shared patterns within its sequences allow the model to effectively learn and generalize to these scenarios. By encompassing a comprehensive range of attack techniques and tactics, CREMEv2 enables the model to match patterns across datasets, enhancing its generalization capabilities.

Moreover, the lifecycle-based dataset is not just a collection of varied attacks; its design aims to closely replicate real-world network conditions compared to conventional attack-based datasets. It captures data from various operational stages of a network, including normal operations, pre-attack indicators, active attack phases, and post-attack scenarios. This comprehensive coverage provides a more accurate reflection of the complexities and nuances found in actual network environments, making the attack data not too specific with the network environment setup.

In addition to the aforementioned investigations, an analysis was conducted on the attack data variety inherent in both the attack-based and lifecycle-based datasets. To visually represent the diversity in data distribution, Principal Component Analysis (PCA) was employed to transform the high-dimensional data into a lower-dimensional representation. PCA aims to find the principal components that capture the maximum variance in the data, and the position of data points on the PCA plot reflects their relative positions in this reduced-dimensional space. The key of PCA is to analyze not only the spread of data but also the direction of maximum variance, which can provide insights into the underlying structure and variability of the data. Figure 7 shows the distribution of the attack data for both CREMEv2



(a) F1 Score



(b) Recall

FIGURE 6. Comparison of generalization performance: lifecycle-based vs. attack-based datasets in F1 Score and recall.

and CIC-IDS-2017. In this visualization, the X- and Y-axes of the scatter plot represent the first and second principal components, while each data point corresponds to a sample extracted from the respective dataset. The precise location of a data point within the scatter plot is determined by its values along PCA1 and PCA2.

The figure clearly illustrates that CREMEv2 data points are distributed across multiple distinct clusters, indicating a variety of underlying patterns, whereas the data points of CIC-IDS-2017 are more tightly grouped, suggesting less variability within the dataset. This suggests that CREMEv2 has a higher level of diversity or variability in its attack data compared to CIC-IDS-2017. This variation in CREMEv2 reflects its ability to capture a wider array of attack behaviors and patterns. This richness in data variety undoubtedly contributes to its heightened generalization performance, allowing the model to effectively recognize the unseen data.

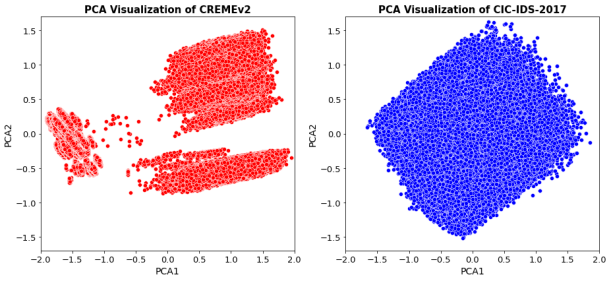


FIGURE 7. Attack Data distribution: CREMEv2 vs. CIC-IDS-2017.

Furthermore, as mentioned earlier, in attack-based datasets such as CIC-IDS-2017, the focus is primarily on generating data based on the impact of the attack. From the perspective of a MITRE framework, these datasets typically revolve around a single tactic. For example, in the case of DDoS attacks, the emphasis is placed on performing a direct network flood on the target.

On the other hand, in the CREMEv2 dataset, which follows a lifecycle-based approach, generating a DDoS using botnet attack involves a sequence of 8 attack techniques, as illustrated in Figure 1. This comprehensive approach serves as a strong factor contributing to its superior generalization capabilities.

An experiment was conducted to demonstrate that the true strength of the CREMEv2 dataset is in its representation of attacks as sequences of techniques within a lifecycle. The goal was to show that when CREMEv2 generates attack data in the same manner as attack-based datasets (emphasizing only the impact of an attack), its capability to effectively generalize unseen data diminishes.

CREMEv2 was configured to use only the 'impact' tactic for DDoS attacks initiated by the botnet. The data was then filtered to keep instances related to the final technique for network flooding and all preceding technique sequences were discarded. Following this, the model's ability to generalize was tested using other DDoS attacks from CIC-IDS-2017, CIC-IDS-2018, and CIC-IDS-2019.

Figure 8 illustrates the generalization performance when the model is trained using only the 'impact' tactic from CREMEv2. It is evident that the model fails to generalize to unseen data during inter-dataset testing, achieving an accuracy rate between 0-0.3. Its performance is primarily effective in classifying data from intra-datasets. This limitation arises because the CREMEv2 dataset becomes overly specific to a particular attack tool, lacking the diversity required for robust generalization.

In conclusion, creating datasets that align with the attack lifecycle is crucial. Detailed representations of an attack's stages enhance dataset variability, improving its capacity to generalize unseen attack data.

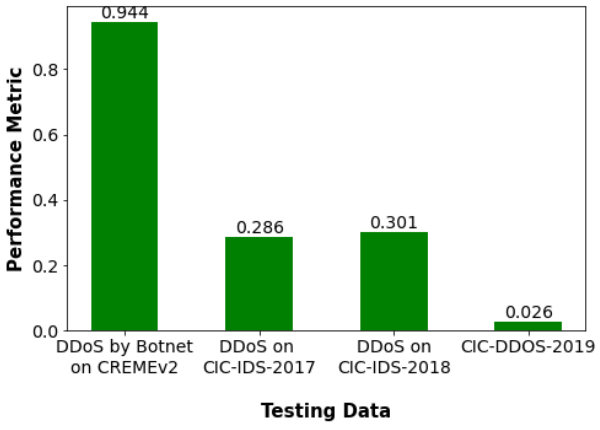


FIGURE 8. Generalization of CREMEv2 only using the 'impact' tactic.

However, despite achieving strong performance in recognizing attack data with a high recall score, CREMEv2 dataset has limitations regarding its benign data. As illustrated in Figure 9, CREMEv2 only attains an average precision score of 0.78, resulting in a relatively high number of false positives compared to false negatives. This indicates that the CNN model might struggle to effectively generalize the benign class, as CREMEv2 only simulates the benign behavior of HTTP, FTP, and SMTP traffic. To address this challenge, novel solutions for generating more comprehensive benign traffic that enhance generalization are needed.

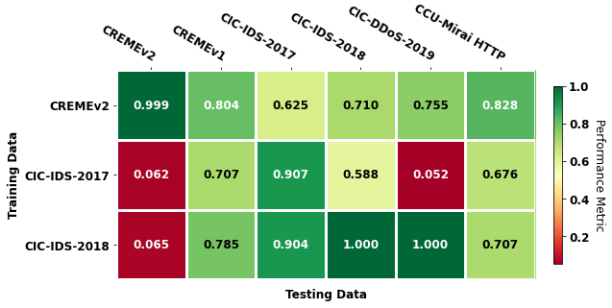


FIGURE 9. Precision score of lifecycle-based vs. attack-based datasets.

C. FEATURES: AUTO-LEARNING Vs. STATISTICAL

In this section, the comparison between the performance of auto-learning and statistical features is explored. The discussion is divided into two main areas: firstly, a direct performance comparison, and secondly, an examination of the key features captured by CNN from the auto-learning process.

PERFORMANCE COMPARISON

A comparison was conducted to assess the impact of different feature extraction methods by training the model on the CREMEv2 dataset. The results, show in Figure 10, highlight the performance of auto-learning features versus statistical features. From the figure, it is evident that all methods achieve high F1 score, exceeding 0.98 when tested with the same training dataset. However, variations in F1 score become apparent when inter-dataset testing is conducted across the methods.

Auto-learning features exhibited a notably higher stability in generalizing the inter-dataset testing when compared to statistical features. The former demonstrated a remarkable ability to generalize across all testing datasets, significantly enhancing overall performance by 0.2. For example, in the case of CREMEv1, the F1 score improved from 0.39 (XGBoost) and 0.61 (CNN) to a substantial 0.84, with an average F1 score of 0.85. This improvement can be attributed to auto-learning's capacity to capture intricate and complex patterns through the automatic extraction of meaningful features from high-dimensional data.

Furthermore, this study also demonstrates that good generalization performance can be achieved despite using only a limited number of packets (3-packets) and a small

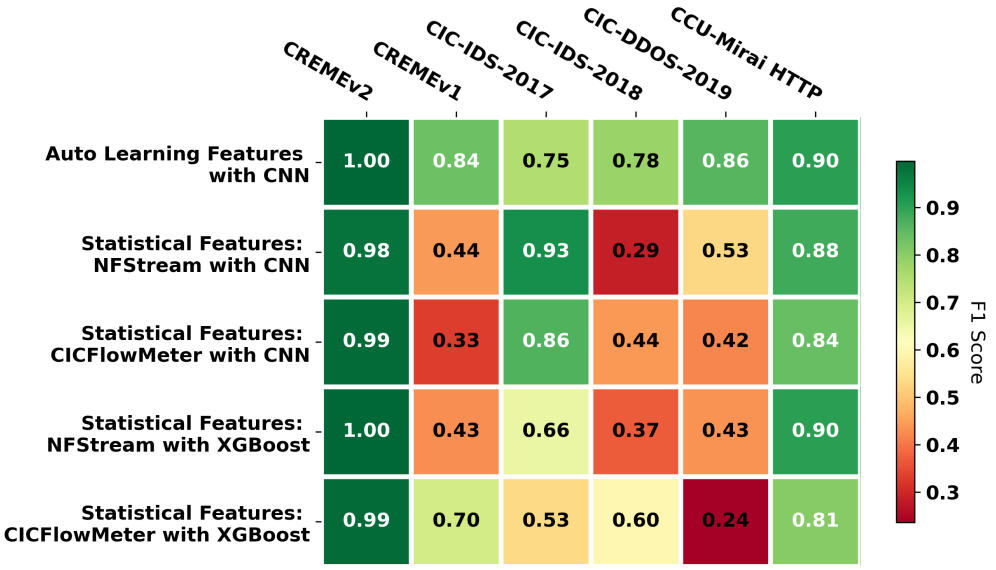


FIGURE 10. **Overview of classification performance of auto-learning vs statistical features trained on CREME v2.**

amount of data (60 bytes from each packet) per flow. The auto-learning feature mechanism employed here utilizes sophisticated analytical techniques that go beyond basic packet analysis. This approach involves scrutinizing data irregularities, such as unusual flags, unexpected packet sizes, or atypical timing. These anomalies are potentially indicative of malicious activities, including scanning, spoofing, or the initial stages of more complex attacks. This broader context provides valuable insights that enable the CNN model to have good generalization.

In contrast, statistical features showcased limitations in their ability to capture complex patterns present in the data. The simplicity of metrics employed, including counts, averages, means, max, and min, hindered their capacity to comprehensively capture crucial data nuances. Consequently, the model's capability to learn intricate data patterns was restricted. Further investigation of statistical features revealed that even when utilized with different statistical feature extractors such as NFStream and CICFlowMeter, the model still struggled to generalize with respect to CREMEv1, self generated brute force, and CIC-IDS-2018.

KEY FEATURES CAPTURED BY AUTO-LEARNING

Furthermore, an investigation was conducted to explore the reasons underlying the impressive recognition capabilities of CNN with auto-learning features trained using CREMEv2. To accomplish this, feature importance scores were calculated by computing the gradients of the model output with respect to the input features. By computing these gradients, it becomes possible to determine which features have the most significant impact on the model's predictions.

The key features present in several datasets, including CREMEv2, CICIDS-2017, and CICIDS-2018, were investigated. The findings reveal that the CNN model considers distinct key features when analyzing each testing dataset, as shown in Figure 11. The ability of the CNN to adapt and learn varied and adaptive representations from different datasets is crucial. This adaptability to the variability of different datasets empowers CNN's generalization performance.

Moreover, CNN's architecture facilitates the construction of feature hierarchies, ranging from basic and local features to intricate and widespread features. This hierarchical approach allows the CNN to capture the varying complexities of network patterns present in each testing dataset. As a result, the CNN's capacity to learn and interpret these intricate patterns contributes significantly to its robust generalization performance.

Figure 11 provides additional insight, showing that the CNN places significant emphasis on feature index 102-106, which corresponds to the third packet within a network flow. This packet furnishes the most pertinent context and information. In a network communication sequence, the first packet serves as the SYN packet initiating the session, followed by the second packet, which is the SYN-ACK response. The third packet, an ACK, marks the commencement of bidirectional data exchange. This sequence encapsulates the distinctive network pattern and lays the foundation for CNN's decision-making process.

In the analysis of these key features, it was found that they are predominantly linked to TCP options information. TCP options encompass additional settings and parameters within the TCP headers of network packets. These options are pivotal in capturing distinct network behaviors and attributes due to their inclusion of additional details about the management of TCP connections, such as how a TCP connection is established, maintained, and terminated, which can vary depending on specific systems, applications, and network environments. These options also provide a way to capture fine-grained details of network interactions, making them as a valuable features for the CNN model.

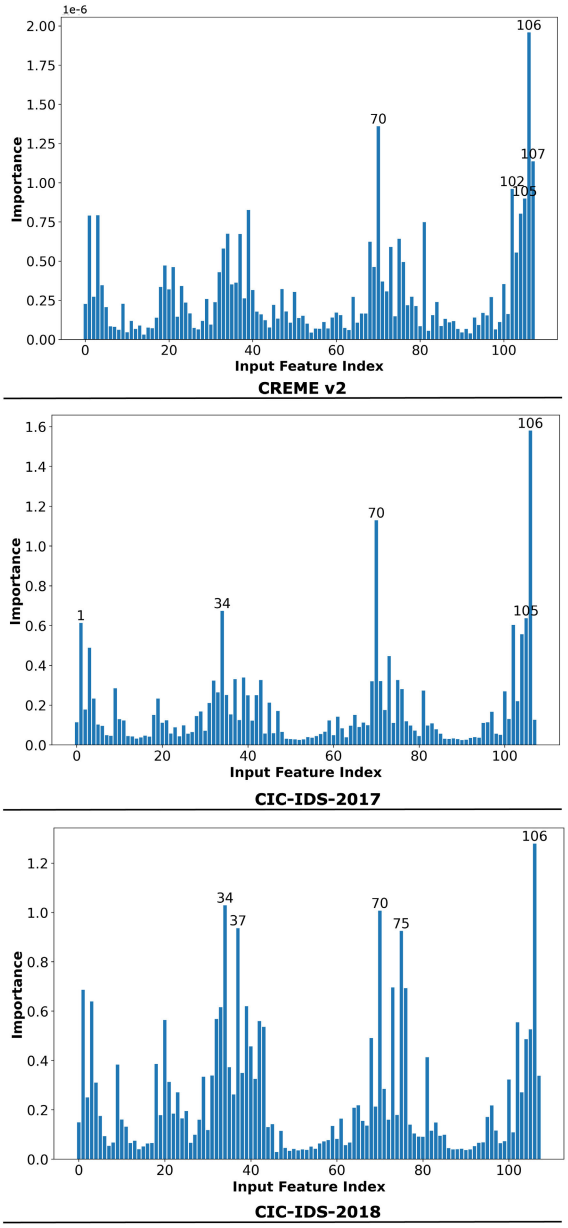


FIGURE 11. Key features computed by model gradients for the CNN model on CREME v2, CIC-IDS-2017, and CIC-IDS-2018.

Furthermore, the distribution of key data features related to the TCP options were investigated, specifically focusing on features 106, 107, 105, and 70. Figure 12 visually presents the data distribution of these features across four distinct datasets. The x-axis represents the value range of specific features within the datasets, while the y-axis illustrates the frequency of occurrence of these values for that particular feature in the dataset. The analysis of the figure reveals that CREMEv2 exhibits more diverse distributions with a wider range of values in these features compared to the other datasets. Utilizing CREMEv2 as the training data, with its diversified feature distributions, can prove advantageous for enhancing generalization performance. By training on a broader spectrum of data patterns, the model develops increased robustness and adaptability to different variations and potential data scenarios. Training on such a diverse dataset equips the model to handle a wider array of patterns, enhancing its adaptability to various variations and potential data scenarios.

Moreover, it also can be seen that there is overlapping data distributions between CREMEv2 and the other datasets. The convergence of distributions between the testing datasets and the training data (CREMEv2) indicates that the model has encountered similar patterns during its training phase. Consequently, when testing data exhibits resemblances in distribution characteristics, the model is better poised to handle such scenarios. This capacity of the model to generalize effectively to data that aligns with its training distribution forms a pivotal aspect of achieving commendable performance.

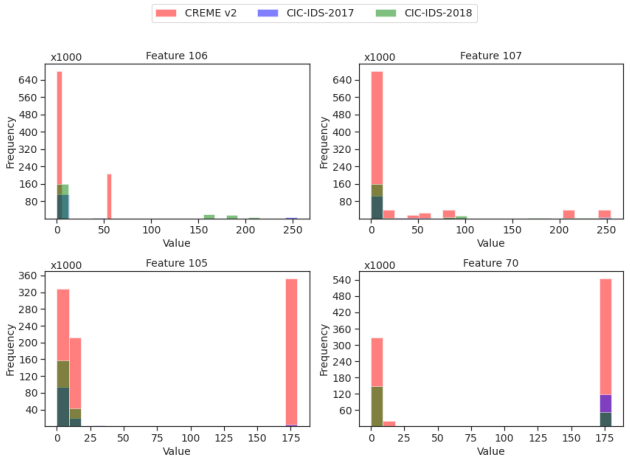


FIGURE 12. The distribution of key data features.

A comprehensive analysis of the TCP options data revealed that the length of the TCP options field can range from 0 to 40 bytes. Notably, this field might not be fully occupied in every packet. The precise length of the options field in a given packet can differ, contingent upon which options are activated and the respective data they encompass. For example, the Maximum Segment Size (MSS) option occupies 4 bytes, while the Timestamps option takes up 10 bytes.

The raw PCAP data in the CREMEv2 dataset was extensively examined to identify the exact values of the TCP option data as essential features. Given the variable nature of this field, only two specific options were extractable: WScale and timestamps. The WScale (Window Scale) option in TCP fields allows for an increase in the maximum window size beyond its original 65,535 bytes. This window size informs the sender how much data the receiver can handle, acting as flow control. In high-latency networks with increasing bandwidth, a larger window size is essential to optimize bandwidth utilization. Additionally, the timestamps option within the TCP options field provides supplementary information about the timing of data transmission and acknowledgments within a TCP connection. Using this

timestamp information, the Round-Trip Time (RTT) for the third packet in each flow was calculated.

The distribution of WScale and RTT values between the benign and attack data of CREMEv2 was then compared, as shown in Figure 13. From the figure, it is evident that both benign and attack data exhibit diverse distributions, enabling the CNN model to effectively distinguish patterns between them. Moreover, there exists a relationship between the WScale data and the RTT. When the RTT is long, having a large window size becomes crucial to maintain uninterrupted data transmission without frequent waits for acknowledgments. The WScale option offers a means to expand this window size, thereby facilitating the full utilization of available bandwidth even in the presence of extended RTTs. As shown in Figure 13, attack traffic exhibits longer RTT values compared to benign traffic, necessitating the use of WScale data. This explains why only 17% of benign data includes WScale data, as benign RTT durations are generally shorter.

However, while this observation is valuable, it is important to note that the CNN model does not heavily rely solely on these values. As Figure 11 illustrates, there are several other combinations of features used by the CNN to comprehensively learn the behavior of CREMEv2 data.

Furthermore, CNNs effectively detect patterns regardless of the order of information in a packet. Their convolutional layers, containing filters (kernels), are crucial for this capability. These filters, designed to identify specific patterns like edges or textures, are applied universally across the input due to shared weights and local connections. This ensures consistent pattern recognition by the network, even with varying feature information order [36].

Finally, the CNN model's ability to detect intrusions with just three packets stems from its training on raw byte representations. It allows it to uncover complex patterns within the data that human analysts or traditional detection systems may overlook. During the initial handshake process, seemingly standard TCP flows offer subtle clues that the CNN model can exploit. For instance, variations in TCP window size, sequence and acknowledgment numbers, and specific TCP options can indicate various types of malicious activities. Through training, the CNN model learns to recognize these associations, leveraging them to identify potential threats accurately. Converting packet data into an image-like format enables the CNN model to operate within a high-dimensional feature space, facilitating the identification and comprehension of intricate relationships between different packet features. This comprehensive analysis, combined with the model's ability to learn hierarchical features from raw byte images, greatly enhances its performance and effectiveness in ML-based IDS.

D. THE EFFECTS OF PARAMETERS

This section explores the comparison between the effects of some parameters on the generalization performance. The discussion is divided into two main areas: firstly, the effect of the number of convolution layers on CNN, and secondly, the effect of packet count and byte size on auto-learning features.

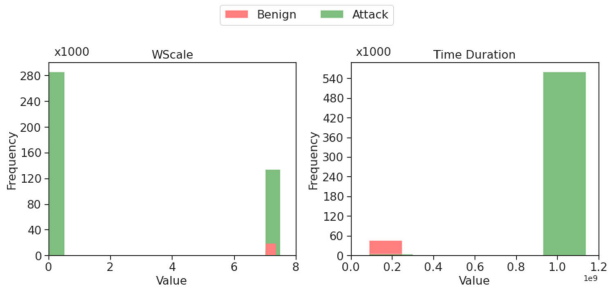


FIGURE 13. The distribution of WScale and RTT values.

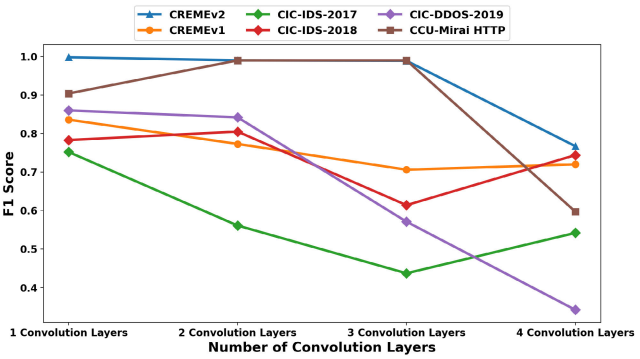


FIGURE 14. Effect of varying convolution layers on generalization performance.

NUMBER OF CONVOLUTION LAYERS ON CNN

An investigation was carried out to assess the impact of convolution layers on generalization performance by augmenting the number of layers in the model architecture. The results, as shown in Figure 14, revealed that an increase in convolution layers actually led to a reduction in generalization performance across several datasets, including CREMEv1, CCU Mirai, CICIDS-2017, and CIC-DDOS-2019. For instance, in the case of CREMEv1, the generalization performance degraded by 0.27, decreasing from 0.93 to 0.76. Similarly, for CICIDS-2017, the degradation was even more pronounced at 0.4, dropping from 0.97 to 0.58.

This phenomenon can be attributed to the hierarchical feature extraction process of convolutional layers. Typically, convolutional layers extract features in a hierarchical manner. The initial layer might detect simple patterns, such as edges in images, while subsequent layers identify progressively more complex structures, utilizing the patterns detected by the earlier layers. Deeper layers delve into even more intricate details.

In the context of IDS, it is conceivable that the most critical and discriminative features for identifying network anomalies are relatively low-level and can be captured by just one convolutional layer. Additional layers might introduce unnecessary complexity or capture overly abstract features that do not significantly contribute to intrusion detection. This can lead to overfitting on the training data, where the network

captures noise instead of general patterns, resulting in poorer generalization when encountering unseen data.

Furthermore, each convolution operation, typically followed by pooling process, reduces the spatial dimensions of the data. This reduction is beneficial to some extent, as it focuses on more abstract and essential features. However, an excessive number of convolution operations poses the risk of the model losing critical information that might be valuable for the IDS task.

PACKET COUNT AND BYTE SIZE ON AUTO-LEARNING FEATURES

An investigation was conducted to assess the impact of modifying the configuration of auto-learning features by including additional packets and bytes. The goal was to discern how changing these parameters affects the auto-learning process.

The findings indicate that increasing the number of packets and bytes detrimentally affects generalization performance, as shown in Figure 15. The datasets most significantly impacted by this change, CIC-IDS-2017 and CIC-DDoS-2019, saw the most severe degradation in performance, dropping from 0.75 and 0.86, respectively, to below 0.4.

Further analysis indicated that attack traffic usually comprises packets with smaller sizes and bytes [37], particularly in the case of DoS or DDoS attacks. Introducing more bytes inadvertently added unnecessary information. This led to increased complexity and dimensionality of the data, which, in turn, made it difficult for the learning process to identify relevant patterns and relationships within the data. As a result, the increase in packets and bytes adversely affects the model's capability to effectively generalize across different testing datasets.

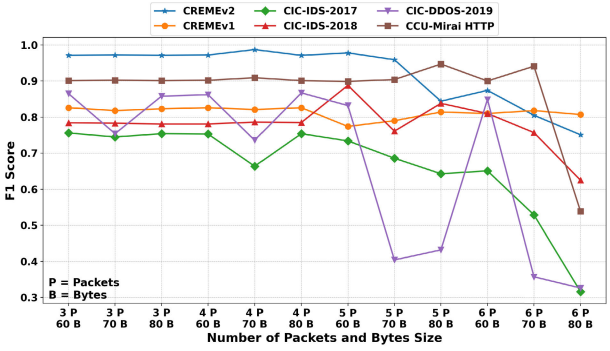


FIGURE 15. Effect of varying packet count and size of auto learning features on generalization performance.

VI. CONCLUSION AND FUTURE WORK

This study aims to enhance model generalization through a novel composite approach: the incorporation of a lifecycle-based dataset, the utilization of auto-learning features, and the employment of a deep learning model. Specifically, the model was trained using a CNN equipped with auto-learning features. The inter-dataset testing strategy is implemented, wherein training and testing are conducted on separate datasets. In total, five public datasets are utilized, and the results demonstrate that the proposed approach can effectively generalize across all testing datasets.

The dataset emerges as the most pivotal factor in boosting generalization, followed by features and the learning models. By employing the lifecycle-based dataset, auto-learning features with CNN, achieve an impressive average F1 score of 0.85 on the testing datasets. This significantly outperforms the best configuration involving attack-based datasets, which yield an average F1 score of 0.45.

The distinction between attack-based and lifecycle-based datasets emerges as a critical consideration. The strategic mapping of attacks to lifecycles in CREMEv2 not only characterizes attacks in terms of techniques and lifecycles but also illuminates the attack's behavioral sequence. The greater variety inherent in this approach infuses it with superior generalization power on attack data.

Comparing the two feature extraction approaches, the findings reveal that auto-learning features exhibit a notable capability to capture intricate patterns directly from raw data, thereby elevating the model's generalization performance. Conversely, statistical features, rooted in simple metrics such as counts and averages, fall short of comprehensively capturing the intricate complexities of the data.

Lastly, from the learning models' perspective, the combination of auto-learning features with CNN is explored. This synergy empowers the model to discern local patterns by extracting insights from smaller data fragments. Moreover, it equips the model to manage variations, making it resilient to outliers or noise. The capability to distinguish different traffic types is especially enhanced when using a single convolution layer.

In conclusion, this study introduces a holistic approach to enhance model generalization by utilizing a lifecycle-based dataset, auto-learning features, and a well-optimized CNN model. The cumulative impact of these efforts yields substantial improvements in model generalization, promising a more robust and adaptive approach to intrusion detection challenges.

Potential future work can include investigating the effects of utilizing multi-data sources, such as traffic, accounting, and Syslog, for model generalization in binary (benign/malign) and multi-class (type of attack) classification. Additionally, exploring the generalization performance by detecting attacks using a two-stage ML process—first detecting the technique and then identifying the lifecycle based on the sequences of techniques—could be a valuable research direction. Another area for future exploration involves improving the generalization of the CREMEv2 dataset regarding benign data. This can be achieved by developing a comprehensive method for generating benign data. Apart from that, incorporating additional attack techniques and lifecycles from the MITRE ATT&CK framework can also be considered. This expansion aims to enable the creation of larger and more varied datasets, thereby enhancing the robustness of machine learning model generalization performance.

REFERENCES

[1] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Jul. 2019.

[2] A. Halbouni et al., "Machine learning and deep learning approaches for cybersecurity: A review," *IEEE Access*, vol. 10, pp. 19572–19585, 2022.

[3] S.-W. Lee et al., "Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review," *J. Netw. Comput. Appl.*, vol. 187, Aug. 2021, Art. no. 103111, doi: 10.1016/j.jnca.2021.103111.

[4] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artif. Intell. Rev.*, vol. 55, pp. 453–563, Jul. 2021, doi: 10.1007/s10462-021-10037-9.

[5] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, Oct. 2020, Art. no. e4150, doi: 10.1002/ett.4150.

[6] S. Dwibedi, M. Pujari, and W. Sun, "A comparative study on contemporary intrusion detection datasets for machine learning research," in *Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI)*, Nov. 2020, pp. 1–6.

[7] L. D'hooge, M. Verkerken, T. Wauters, F. De Turck, and B. Volckaert, "Characterizing the impact of data-damaged models on generalization strength in intrusion detection," *J. Cybersecurity Privacy*, vol. 3, no. 2, pp. 118–144, Apr. 2023, doi: 10.3390/jcp3020008.

[8] A. Thakkar and R. Lohiya, "Analyzing fusion of regularization techniques in the deep learning-based intrusion detection system," *Int. J. Intell. Syst.*, vol. 36, no. 12, pp. 7340–7388, Aug. 2021, doi: 10.1002/int.22590.

[9] L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Inter-dataset generalization strength of supervised machine learning methods for intrusion detection," *J. Inf. Secur. Appl.*, vol. 54, Oct. 2020, Art. no. 102564.

[10] L. D'hooge, M. Verkerken, T. Wauters, F. De Turck, and B. Volckaert, "Investigating generalized performance of data-constrained supervised machine learning models on novel, related samples in intrusion detection," *Sensors*, vol. 23, no. 4, p. 1846, Feb. 2023.

[11] M. Verkerken, L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Towards model generalization for intrusion detection: Unsupervised machine learning techniques," *J. Netw. Syst. Manage.*, vol. 30, no. 1, p. 12, Oct. 2021.

[12] Z. Yang et al., "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Comput. Secur.*, vol. 116, May 2022, Art. no. 102675, doi: 10.1016/j.cose.2022.102675.

[13] G. Kocher and G. Kumar, "Machine learning and deep learning methods for intrusion detection systems: Recent developments and challenges," *Soft Comput.*, vol. 25, no. 15, pp. 9731–9763, Jun. 2021.

[14] J. Lansky et al., "Deep learning-based intrusion detection systems: A systematic review," *IEEE Access*, vol. 9, pp. 101574–101599, 2021.

[15] F. Yudha. (2023). *Cremev2: A Toolchain of Automatic Dataset Collection for Machine Learning in Intrusion Detection Based on Mitre Att&CK*. [Online]. Available: https://github.com/masjohncook/CREMEv2

[16] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," *IEEE Access*, vol. 8, pp. 30387–30399, 2020.

[17] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI)*, Jul. 2017, pp. 43–48.

[18] M. Monshizadeh, V. Khatri, M. Gamdou, R. Kantola, and Z. Yan, "Improving data generalization with variational autoencoders for network traffic anomaly detection," *IEEE Access*, vol. 9, pp. 56893–56907, 2021.

[19] M. A. Bouke and A. Abdullah, "An empirical study of pattern leakage impact during data preprocessing on machine learning-based intrusion detection models reliability," *Expert Syst. Appl.*, vol. 230, Nov. 2023, Art. no. 120715.

[20] Z. Zeng, W. Peng, and D. Zeng, "Improving the stability of intrusion detection with causal deep learning," *IEEE Trans. Netw. Service Manage.*, vol. 19, no. 4, pp. 4750–4763, Dec. 2022.

[21] G. de Carvalho Bertoli, L. A. P. Junior, O. Saotome, and A. L. dos Santos, "Generalizing intrusion detection for heterogeneous networks: A stacked-unsupervised federated learning approach," *Comput. Secur.*, vol. 127, Apr. 2023, Art. no. 103106.

[22] L. H. de Melo, G. de C. Bertoli, L. A. Pereira, O. Saotome, M. F. Domingues, and A. L. dos Santos, "Generalizing flow classification for distributed denial-of-service over different networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2022, pp. 879–884.

[23] R. Magán-Carrión, D. Urda, I. Diaz-Cano, and B. Dorronsoro, "Improving the reliability of network intrusion detection systems through dataset integration," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 4, pp. 1717–1732, Oct. 2022.

[24] M. Marvi, A. Arfeen, and R. Uddin, "A generalized machine learning-based model for the detection of DDoS attacks," *Int. J. Netw. Manage.*, vol. 31, no. 6, p. e2152, 2021.

[25] H.-K. Bui, Y.-D. Lin, R.-H. Hwang, P.-C. Lin, V.-L. Nguyen, and Y.-C. Lai, "CREME: A toolchain of automatic dataset collection for machine learning in intrusion detection," *J. Netw. Comput. Appl.*, vol. 193, Nov. 2021, Art. no. 103212.

[26] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Comput.*, vol. 24, no. 3, pp. 1999–2012, May 2019.

[27] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4707749

[28] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Security Technol. (ICCST)*, 2019, pp. 1–8.

[29] L. Liu, G. Engelen, T. Lynar, D. Essam, and W. Joosen, "Error prevalence in NIDS datasets: A case study on CIC-IDS-2017 and CSE-CIC-IDS-2018," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2022, pp. 254–262.

[30] SecDev. (2023). *Scapy: The Python-Based Interactive Packet Manipulation Program & Library*. [Online]. Available: https://github.com/secdev/scapy

[31] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Tor traffic using time based features," in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*. Porto, Portugal: SciTePress, Feb. 2017, pp. 253–262.

[32] Z. Aouini and A. Pekar, "NFStream: A flexible network data analysis framework," *Comput. Netw.*, vol. 204, Jun. 2022, Art. no. 108719. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621005739

[33] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *J. Big Data*, vol. 7, no. 1, Nov. 2020, doi: 10.1186/s40537-020-00379-6.

[34] P. Devan and N. Khare, "An efficient XGBoost–DNN-based classification model for network intrusion detection system," *Neural Comput. Appl.*, vol. 32, no. 16, pp. 12499–12514, Jan. 2020, doi: 10.1007/s00521-020-04708-x.

[35] D. Sudyana. (2024). *Github Repository Containing Code for the Paper 'Improving Generalization of Ml-Based Ids With Lifecycle-based Dataset, Auto-learning Features, and Deep Learning'*. [Online]. Available: https://github.com/nycu-hsl/improving-generalization-of-ML-based-IDS

[36] L. Alzubaidi et al., "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[37] B. Nugraha and R. N. Murthy, "Deep learning-based slow DDoS attack detection in SDN-based networks," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2020, pp. 51–56.

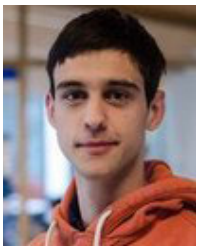
DIDIK SUDYANA received the M.S. degree in informatics from Universitas Islam Indonesia (UII), Indonesia, in 2016. He is currently pursuing the Ph.D. degree in electrical engineering and computer science (EECS) international graduate program with National Yang Ming Chiao Tung University (NYCU). He is a Lecturer of informatics with Universitas Sains dan Teknologi Indonesia (USTI), Indonesia. His research interests include cybersecurity, machine learning, and network design and optimization.

YING-DAR LIN (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California at Los Angeles (UCLA), in 1993. He was a Visiting Scholar with Cisco Systems, San Jose, from 2007 to 2008, the CEO of Telecom Technology Center, Taiwan, from 2010 to 2011, and the Vice President of National Applied Research Labs (NARLabs), Taiwan, from 2017 to 2018. He cofounded L7 Networks Inc., in 2002, and O'Prueba Inc., in 2018. He is currently the Chair Professor of computer science with National Yang Ming Chiao Tung University (NYCU), Taiwan. His research interests include cybersecurity, wireless communications, network softwarization, and machine learning for communications. He served or is serving on the Editorial Boards for several IEEE journals and magazines, including the Editor-in-Chief for IEEE COMMUNICATIONS SURVEYS AND TUTORIALS (COMST, 2017–2020).

MIEL VERKERKEN received the M.Sc. degree in information engineering technology, in 2018. He is currently pursuing the Ph.D. degree with the Internet and Data Science Lab (IDLab-imec), Ghent University. He has been a Teaching Assistant with the IDLab-imec, Ghent University, since September 2019. After the M.Sc. degree, he gained some international and professional experience, before starting as a Researcher. His current research interests include the enhancement of cybersecurity through ML, more specifically applying AI to intrusion detection systems.

REN-HUNG HWANG (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Massachusetts, Amherst. He was the Dean of the College of Engineering, from 2014 to 2017. He is currently the Dean of the College of Artificial Intelligence, National Yang Ming Chiao Tung University (NYCU), Taiwan. Before joining NYCU, he was with National Chung Cheng University, Taiwan, from 1993 to 2022. He has published more than 250 international journal and conference papers. His current research interests include deep learning, wireless communications, network security, and cloud/edge/fog computing. He received the IEEE Best Paper Award from IEEE UbiMedia 2018, IEEE SC2 2017, and IEEE IUCC 2014.

YUAN-CHENG LAI received the Ph.D. degree from the Department of Computer and Information Science, National Chiao Tung University, in 1997. He joined as the Faculty Member of the Department of Information Management, National Taiwan University of Science and Technology, in August 2001, and has been a Distinguished Professor, since June 2012. His research interests include performance analysis, software-defined networking, wireless networks, and the IoT security.



LAURENS D'HOOGE received the M.Sc. degree in information engineering technology from Ghent University, in 2018, and the Ph.D. degree from the Internet and Data Science Lab (IDLab-imec), Ghent University, in 2023. His research interests include the intersection of cybersecurity, more specifically network security and applied machine learning.

TIM WAUTERS (Member, IEEE) received the M.Sc. and Ph.D. degrees in electro-technical engineering from Ghent University, in 2001 and 2007, respectively. He was a Postdoctoral Fellow with FWO-V, Department of Information Technology (INTEC), Ghent University. He is currently active as a Senior Researcher with imec. His work has been published in more than 150 scientific publications. His research interests include design and management of networked services, covering multimedia distribution, cybersecurity, big data, and smart cities.

BRUNO VOLCKAERT (Senior Member, IEEE) received the Ph.D. degree in resource management for grid computing from Ghent University, in 2006. He is currently a professor of advanced distributed systems with Ghent University, and a Senior Researcher with imec. He has been involved in over 45 national and international research projects and is the author or the coauthor of more than 150 peer-reviewed papers published in international journals and conference proceedings. His current research interests include reliable and high performance distributed software systems for a.o. smart cities, scalable cybersecurity detection and mitigation architectures, and autonomous optimization of cloud-based applications.

FILIP DE TURCK (Fellow, IEEE) leads the Network and Service Management Research Group, Ghent University, Belgium, and imec. He has coauthored over 700 peer-reviewed articles. He is involved in several research projects with industry and academia. His research interests include design of secure and efficient softwarized network and cloud systems. He was elevated as an IEEE Fellow for outstanding technical contributions. He served as the Chair for the IEEE Technical Committee on Network Operations and Management (CNOM) and a Steering Committee Member for the IFIP/IEEE IM, IEEE/IFIP NOMS, IEEE/IFIP CNSM, and IEEE NetSoft conferences.