IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 29 March 2025, accepted 18 April 2025, date of publication 2 July 2025, date of current version 11 July 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3585445

RESEARCH ARTICLE

Adaptive Defense: Zero-Day Attack Detection in NIDS With Deep Reinforcement Learning

KHORSHED ALAM[1], MD FAHAD MONIR[1], (Member, IEEE), MD JUNAYED HOSSAIN[1], MOHAMMAD SHORIF UDDIN[2], (Senior Member, IEEE), AND MD. TAREK HABIB[1]

[1]Department of Computer Science and Engineering (CSE), Independent University at Bangladesh, Dhaka 1229, Bangladesh
[2]Department of Computer Science and Engineering, Jahangirnagar University, Dhaka 1342, Bangladesh

Corresponding author: Md. Tarek Habib (md.tarekhabib@yahoo.com)

ABSTRACT Zero-Day attack detection in Network Intrusion Detection Systems (NIDS) refers to the ability to identify previously unseen attack patterns during testing without having been explicitly trained on those specific attacks, utilizing learned features from other known attacks. In this paper, we propose a Deep Reinforcement Learning (DRL)-based NIDS designed for Zero-Day attack detection. We use a stacked LSTM architecture to extend the learning capabilities of the DRL agent. We apply several oversampling techniques to handle the issue of class imbalance since the zero-day attack datasets are not as abundant. We use some of the most widely available benchmark datasets in NIDS domain, which all together cover a wide range of attack types, such as reconnaissance, ddoS, infiltration, injection, password attacks, brute force, dos, backdoor, and benign traffic. For example, we converted attacks to 1 and benign traffic to 0, then excluded certain attack categories (DoS and Backdoor) from the training dataset while keeping them in the test dataset. This makes those attack types zero-day attacks, as they are entirely unseen during training. We also compare which data balancing technique works better among K-means SMOTE, SMOTE, Borderline-SMOTE and ADASYN on the performance of our DRL agent. We then demonstrate how powerful our agent is by validating many datasets for remarkable success in detecting both known and unknown attacks in a zero-day manner. Our work has been made publicly available on GitHub (https://github.com/codewithkhurshed/ZDAD) to support researchers in advancing zero-day attack detection in NIDS.

INDEX TERMS Zero-day attack detection, deep reinforcement learning, cybersecurity, network intrusion detection systems, internet, unseen attack generalization, digital infrastructure, industrialization, governance, SDG 9, SDG 16.

I. INTRODUCTION

Zero day Attack Detection represents one of the state-of-the-art approaches for network intrusion detection system (NIDS), aiming at detecting attacks that were not seen or not trained for. Traditional models in NIDS are based on the use of supervised learning techniques [1], which require large datasets related to known types of attacks that are labeled. However, with the ever-changing face of cyber threats, one major challenge these systems present is that of new attack types or novelties, as systems mostly fail to identify trends they do not experience history for. Zero-day attack detection is therefore to deal with this, trying to make the NIDS detect abnormal patterns that appear deviant from normal benign traffic rather than seeking a specific known attack. This makes the system more adaptable and resilient, generalizes patterns of malicious behavior to novel scenarios for which it has little to no prior exposure.

Developing a network intrusion detection system capable of identifying zero day attacks holds significant relevance in the current cybersecurity landscape, where novel and sophisticated attack methods are constantly being developed [2]. A model with zero day detection capability adds a proactive layer to cybersecurity defenses, reducing the risk of damage from unknown threats. As new attacks emerge at a rapid pace, manually labeling and retraining models for each new

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

attack type becomes impractical and resource-intensive. Zero day detection systems address this by leveraging generalized representations of attack behavior that enable the model to infer and classify previously unseen attack types based on behavioral characteristics and anomaly signals. Such systems not only enhance an organization's capacity to respond to emerging threats but also improve detection coverage, minimizing the attack surface exposed to zero-day. Moreover, these models can streamline response efforts, providing security teams with alerts about novel threats that conventional models might overlook, thus improving the efficiency and efficacy of network security operations. From latest research trends, learning methods like deep learning are essential in scientific research as they enable the analysis of complex, high-dimensional data, uncovering patterns and insights that traditional methods cannot. These techniques drive innovation across fields such as healthcare, cybersecurity, NLP, agriculture and climate science, providing accurate predictions and solutions to real-world problems. In network security, deep learning is particularly relevant for detecting anomalies, identifying intrusion patterns, and building robust defenses against evolving cyber threats.

However, traditional deep learning-based approaches have shown a limited ability to detect zero day attacks, but this capability is constrained by their dependency on labeled datasets and lack of inherent adaptability [3]. DL models trained on historical attack data can sometimes extrapolate behaviors that suggest an attack, even if the exact attack vector was not seen during training. However, this generalization capability is limited, as traditional DL models tend to perform poorly when the data distribution of new attack types deviates significantly from those previously encountered. In contrast, deep reinforcement learning (DRL) offers a more adaptive solution for detecting zero day attacks due to its ability to learn through environmental interaction rather than purely from static, labeled data [4]. DRL agents are trained to optimize a policy based on rewards for correct actions, allowing them to adjust dynamically to new patterns, rather than relying solely on existing categories or data points. DRL's exploration-exploitation framework enables it to detect outlying behaviors and anomalous patterns without requiring predefined labels or static boundaries, making it inherently better suited to handle variations beyond traditional data categories. Consequently, DRL-based systems are more flexible, resilient, and effective in identifying zero day attacks, as they can adaptively learn from interactions in real-time and respond to novel threats that lie outside the conventional boundaries of previously labeled data. This adaptability gives DRL a distinct advantage in cybersecurity applications where threat patterns are highly dynamic and continuously evolving.

This is the new turning of cybersecurity, where the adaptability to novel threats is crucial, given the remarkable performance of NIDS detection applying DRL-based network intrusion detection [5], [6], [7]. Unlike conventional models, which mostly depend on historical data with predefined attack patterns, a DRL-based NIDS learns continuously through dynamic interactions within the network and has the ability to detect and respond to unseen attack types in real time. This allows the NIDS to be more capable of detecting anomalies that would normally not be visible by traditional deep learning methods, especially in cases when new attack vectors are not very similar to known threat signatures.

Practically, a DRL-based zero day NIDS model works by observing the behaviour of the traffic and optimizing its detection strategies via rewards tied to correctly identifying malicious activity. With the self-improvement mechanism, thus inherent to DRL, it provides a great advantage by automatically adapting to the continuously changing threat landscapes without frequent retraining or updates. Such models prove to be very useful in an environment with high variability of traffic, for example, in enterprise networks where new devices, applications, and configurations create diverse patterns-a fixed-model approach might struggle to interpret. By detecting abnormal behaviors without being previously exposed to attack types, the DRL-based zero-day NIDS models reduce the likelihood of hidden breaches, thus playing an important role in proactive security measures and rapid response. The flexibility of network defense thus allows security teams to allocate resources and strategic mitigation, rather than performing routine and time-consuming model updates-a facilitator of robust, efficient, and proactive cybersecurity practices.

Our work proposed a stacked long short term memory (LSTM) based architecture as DRL agent for zero day attack detection in NIDS. List of our contribution is stated bellow:

- **Enhanced Learning Framework:** Utilized a stacked LSTM architecture to improve the learning capabilities of the DRL agent.
- **Zero-Day Training Strategy:** Developed a unique training approach where some types of attack data were excluded from the training set and reserved for testing, treating them as zero-day attacks. For example, in the NF-BoT-IoT dataset [8], we removed the DoS and theft attack categories from the training set and included them in the testing set. Although the agent was not exposed to any data from these attack categories during training, it was still able to successfully detect these intrusions during testing, as discussed in Section V.
- **Data balancing Strategy:** We conducted a comparative analysis of various data balancing techniques (SMOTE, Borderline-SMOTE, ADASYN, and K-means SMOTE) to identify the most effective methods for enhancing agent performance. This evaluation provides valuable guidance for future researchers in selecting the most suitable balancing techniques for the development of NIDS.
- **Robust Validation:** Demonstrated the robustness and effectiveness of the proposed DRL agent through multi-dataset validation, achieving significant success in detecting both known and unknown attacks.

In Section II, we present a contextual overview of topics relevant to our research. Section III covers prior studies, while Sections IV and V detail the methodology and experimental results. Section VI offers a discussion, Section VII outlines limitations, Section VIII suggests future work, and Section IX concludes the research.

II. BACKGROUND

In this section, we offer a contextual overview of the topics relevant to our research, aimed at readers from disciplines outside of computer science and network security.

A. NETWORK INTRUSION DETECTION SYSTEM (NIDS)

An NIDS [9] is a security solution to monitor and analyze network traffic for potentially threat activity or violations of security policy shown in FIGURE 1. This works through the examination of data packets across a network utilizing different detection methods: signature-based detection-means it identifies the known threats through predefined patterns; and anomaly-based detection-establishes a baseline of normal network use and flags deviations from that norm. It can also alert the network administrators in case of a recognized potential threat for an immediate response that must help in preventing serious damage. This will make NIDS very important in protecting network infrastructure from cyber attacks, unauthorized access, and other security breaches through real-time monitoring and analysis.

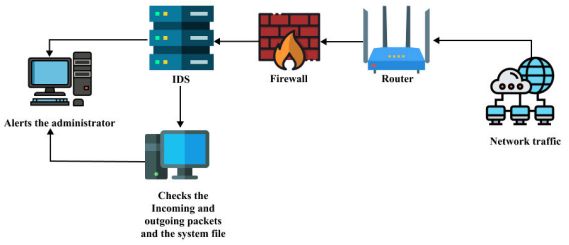


FIGURE 1. Network intrusion detection system (NIDS).

B. ZERO-DAY ATTACK DETECTION IN NIDS

Zero-Day attack detection in Network Intrusion Detection Systems refers to the ability of the system to identify and respond to attack types that it has never encountered during training [10]. Traditional NIDS typically rely on learning from labeled datasets containing known attack types, which limits their capability to detect new or emerging threats. In contrast, zero-day detection allows a model to recognize unfamiliar attacks by leveraging patterns learned from other types of known attacks, enabling it to generalize its understanding of malicious behavior. ''zero-day'' means the model has ''zero examples'' of the specific attack type in its training data.

To implement zero-day detection, we developed a unique training approach where specific attack types were deliberately excluded from the training dataset and reserved exclusively for the testing phase shown in FIGURE 2.

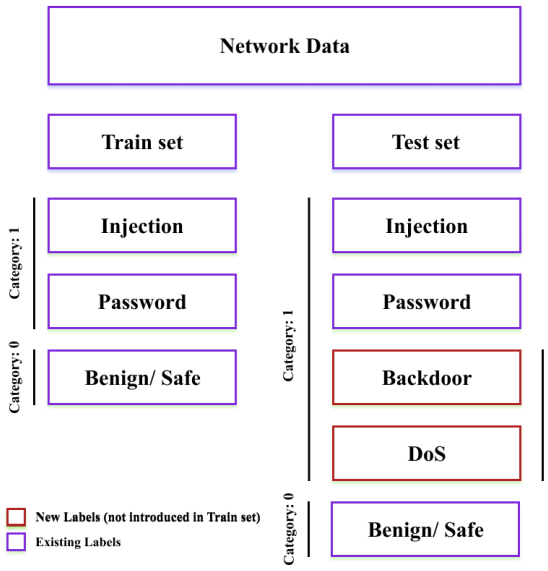


FIGURE 2. Zero-day attack detection training strategy.

For example, in the NF-BoT-IoT dataset, we removed the DoS and backdoor attack categories from the training set, so the model was not exposed to any instances of these attacks during training. These types were then included in the testing set, allowing us to assess the model's zero-day detection capabilities. Despite having no prior exposure to these categories, the model successfully detected the DoS and backdoor intrusions during testing. This training strategy demonstrates the potential of our model to detect novel attacks without requiring explicit data for every possible threat, addressing the growing challenge of zero-day vulnerabilities in network security.

C. DEEP REINFORCEMENT LEARNING (DRL)

DRL combines the decision-making power of reinforcement learning with the pattern recognition capabilities of deep learning, making it a powerful tool for complex tasks like zero-day attack detection in Network Intrusion Detection Systems [11], [12]. In DRL, an agent learns to make decisions by interacting with an environment, receiving feedback through rewards or penalties that guide it toward optimal behavior shown in figure 3. Key components of DRL include the agent, which performs actions; the environment, where the agent operates; and the reward function, which evaluates the agent's actions, encouraging successful behavior over time. Unlike traditional deep learning models that require large, labeled datasets with explicit examples of every attack type, DRL agents learn dynamically through exploration and adapt by generalizing learned features across different attack types. This adaptability is particularly advantageous for zero-day attack detection, where the model needs to recognize new, unseen threats. By training the DRL agent on varying patterns of network behavior, it becomes adept at identifying anomalies indicative of potential attacks—even

those it hasn't encountered before—offering a more flexible and resilient approach than traditional DL methods.

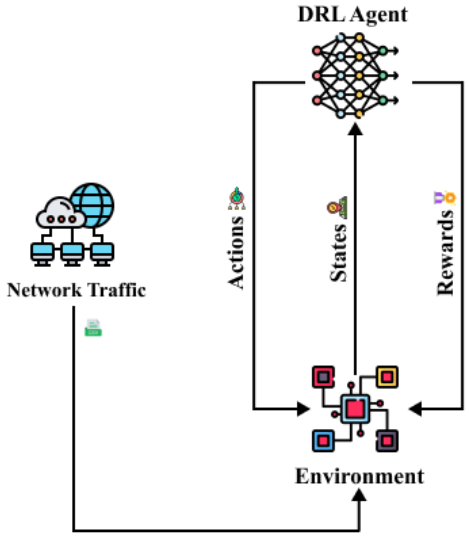


FIGURE 3. Deep Reinforcement Learning Workflow in NIDS.

D. ROLE OF ZERO DAY ATTACK DETECTION BASED NIDS

Zero-day attack detection is of great importance in modern cybersecurity, where contemporary networks become more and more complex and adversaries develop really sophisticated ways of carrying out attacks [13], [14]. Traditional approaches to NIDS rely on very large labeled datasets of all known types of attacks to detect threats with a high quality, which turns them ineffective against attacks they have not seen before. Zero-day detection mitigates this limitation, hence allowing NIDS to find completely new and emerging threats without prior exposure. It has wide applications across sectors that heavily rely on secure networks. The key applications of zero-day NIDS are in protecting critical infrastructure such as energy grids, government communication channels, and financial networks. Security breaches in such environments may have economic, political, and operational consequences. A zero-day NIDS provides detection by anomaly and potential threat against standard patterns without requiring labeled data for each and every type of attack. It allows for the swift detection of the attack methods that are only just emerging and gives the security team an opportunity to respond before the attacks can cause serious harm.

For corporate and enterprise setups, zero-day NIDS serves as a very worthy solution to protect everything from intellectual property to sensitive customer data, even extending to the protection of organizational operations [30], [31], [32]. Considering the rise of advanced persistent threats (APTs) and insider attacks, companies require systems that can adapt faster to newer and changing tactics. Zero-day NIDS diminishes the need for the continuous retraining of emerging threats in detection models-a time-consuming and expensive task. Zero-day learning enables an organization to keep security robust at times when the attack vectors are usually changing fast, hence increasing the general resilience of the network with less downtime or loss of data due to undetected breaches. The need for zero-day NIDS is even more pronounced in the Internet of Things (IoT) domain, where a high number of interconnected devices generate massive amounts of data, making it challenging to label attack types comprehensively. Many IoT devices lack strong built-in security and operate in environments where attacks can easily go undetected by traditional methods. Zero-day NIDS enables IoT networks to detect potential attacks without having prior examples of each attack type, which is essential for ensuring the security of connected devices in smart homes, healthcare, and industrial IoT applications. The application of Zero-day NIDS is also quite impressive in threat intelligence and situation awareness. Within those environments where NIDS feeds into other broader threat intelligence systems, zero-day allows fast flagging of unknown threats that can then be analyzed to understand the emerging attack trends and techniques. Such information has the ability to accelerate decision-making, proactive defenses that fortify an organization's resilience toward new attack vectors before they become widespread.

III. RELATED WORK

NIDS are a continuously growing research area. Based on certain demands and applications, many methods have been proposed by scientists in the improvements of NIDS. Among the mainstream approaches where traditionally DL is superior to handle known attack types, zero-day attacks are seldom seen while detecting an attack type that has not been seen before. On the other hand, some researchers create a more adaptive defense by developing reinforcement learning-based methods, which may be more effective in the detection or mitigation of zero-day or unknown attacks, thus providing stronger protection for network environments. RL has shown significant potential in detecting anomalous network access and countering various forms of network attacks, as highlighted by Benaddi et al. [15]. Further studies, such as those by [16] and [17], demonstrate the effectiveness of deep reinforcement learning in addressing complex security challenges. Modern NIDS encounter substantial difficulties in identifying advanced and evolving cyber threats, as traditional methods often struggle with processing large volumes of network data and managing complex traffic patterns. These limitations contribute to high false negative rates and reduced efficacy in recognizing new attack types. While machine learning-based methods, such as network packet classification, have advanced network security performance, they still face challenges in handling large-scale, complex data and detecting sophisticated attack patterns.

Al-Fawa'reh et al. [18] proposed MalBoT-DRL, the first botnet detection model with the ability to adapt dynamically to the evolving malware pattern, powered by DRL. On the early and later detection phases, the effectiveness

of MalBoT-DRL has been demonstrated on MedBIoT and N-BaIoT datasets with accuracy values of 99.80% and 99.40%, respectively. However, it still faces the challenge of detection for stealthier malware like Bashlite, is susceptible to adversarial attacks, and higher latency for processing larger datasets. Despite many limitations, the model always shows training efficiency. Therefore, future work can work on finding better exploration-exploitation strategies and may be used to perform network distillation for alleviating the above-mentioned challenges. This will ensure further enhancement of the robustness with minimum computational overhead in practical IoT scenarios. DRL-based approach showed a promising result, authors He et al. [19] proposed a transferable and adaptable network intrusion detection system, TA-NIDS, which leverages deep reinforcement learning. Their approach improves robustness and adaptability by exploiting small-scale datasets to generate various interactions, prioritizes outliers without necessarily classifying the complete dataset, and guarantees model transferability across different datasets. The experimental results showed high accuracy with effective prioritization of outliers. Ma and Shi [20] presented a novel framework for anomaly detection in IDS. This was achieved by incorporating RL with class-imbalance techniques, which gave an added boost to the detection accuracy of the model. In this process, the adapted Synthetic Minority Over-sampling Technique, referred to as SMOTE, is applied to deal with dataset imbalance issues, thereby improving the performance of the RL agent. The experiments on the NSL-KDD dataset show the model's performance. Comparative studies have evidenced that the proposed model outperforms AE-RL, with an accuracy of over 0.82 and an F1 score of over 0.824. Limitations include dependence on a single dataset, the NSL-KDD, which itself is not representative of every real-world scenario. Multi-dataset validation can be used to prove the robustness in terms of real-world scenarios of the proposed solution.

A growing body of research highlights the significant advancements in applying GANs to intrusion detection systems. The work in [21] introduced a projection-based adversarial attack generation for IDS using a traffic space GAN to model the distribution of malicious and benign traffic. Similarly, [22] applied GANs to generate synthetic data, followed by classification using deep neural networks, convolutional neural networks (CNN), and long short-term memory, demonstrating that the GAN+DNN combination provided superior performance on the UNSW-NB15 dataset. Studies like [23] have addressed challenges in GAN training, such as instability and mode collapse, by proposing ensemble-based multi-layer GANs (EMP-GANs) that enhance training stability and synthetic data diversity. Meanwhile, traditional AI methods like decision trees (DTs) and support vector machines have been widely explored in early network intrusion detection efforts. A comparative study in [24] evaluated Naïve Bayes, SVM, and K-Nearest Neighbors on NSL-KDD and UNSW-NB15 datasets, revealing varied results in intrusion detection accuracy across algorithms. The study [25] uses zero-day learning to adapt a completely new approach in the case of network attack detection of unknown types. It solves problems that traditional approaches fail to solve: they require labeled data in huge amounts and often offer very poor accuracy when it comes to attacks of types unknown to the system. The authors introduce a variational autoencoder-based ZSL model that tries to align the semantic and visual embeddings in a multimetric space, enabling knowledge transfer from known to unknown attack classes via semantic feature vectors. This helps avoid the projection domain shift problem very prevalent in ZSL and enhances detection accuracy for unfamiliar attacks.

Authors from [26] addresses the challenge of detecting zero-day attacks in the Internet of Vehicles (IoV) domain, where limited labeled attack data makes traditional detection methods unreliable. The authors propose an innovative few-day learning approach using a conditional GAN with multiple generators and discriminators to tackle this issue. By implementing an adaptive sampling data augmentation method, the framework enhances known attack samples to reduce false positives. Additionally, a collaborative focus loss function is introduced to mitigate data imbalance, prioritizing challenging cases in the classification process. The proposed model's efficacy is validated through extensive testing on the F2MD vehicle network simulation platform, where it demonstrated superior detection accuracy and reduced latency compared to existing methods. These results highlight the method's practical potential as a robust solution for zero-day attack detection in IoV environments. However, the approach may face challenges in real-time implementation due to the complexity of the conditional GAN framework, which could lead to increased computational overhead and latency in dynamic IoV environments.

The study [27] introduces a novel approach utilizing LSTM algorithms, which excel in capturing temporal data correlations to simulate and identify these threats. The project establishes a sophisticated framework designed to model complex zero-day attacks that emulate previously undiscovered vulnerabilities. By leveraging recurrent neural networks and advanced gating techniques, the LSTM-based detection method showcases its ability to recognize new attack patterns and detect subtle deviations from normal behavior. Through rigorous testing, the system demonstrates enhanced accuracy and responsiveness, ultimately mitigating the potential damage posed by unknown vulnerabilities. This innovative approach represents a significant advancement in cybersecurity, marking a paradigm shift in how zero-day threats are addressed. Authors from [28] proposes a zero-day deep learning approach for Botnet detection, utilizing a Deep Sparse Contrastive Auto-Encoder (DSCAE) model that enhances detection performance through meaningful latent feature representation. The proposed model was tested against classical machine learning methods and a standard Auto-Encoder, with results indicating that the DSCAE model

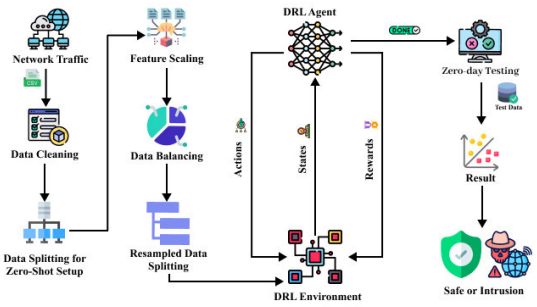


FIGURE 4. Methodology diagram of our NIDS.

consistently improves detection metrics and often achieves the highest performance compared to other approaches.

In the paper [29], the authors introduce the ''Zérosdetect'' approach: a novel quantum-powered zero-shot learning-based method for phishing URL detection. Such a powerful model leverages the flexibility of zero-shot learning with computational power from quantum computing to limit the amount of prior information about the phishing URLs. Such systems use quantum neural networks deployed to transform URL data into quantum spaces, enabling them to make use of the high computational power of quantum systems to discover phishing attacks that were not detected before. It includes quantum layers in Q-nodes, enabling fast and efficient gradients that help in optimizing network performance; hence, the solution is effective and highly innovative. In general, this research provides a sound basis for various cybersecurity undertakings in the quantum computing era, and enormous improvement in comparing capability for the prediction and prevention of new types of phishing attacks. R. Desai and T.G. Venkatesh [33] introduces a Robust Network Intrusion Detection System (RNIDS) that leverages a Convolutional Neural Network (CNN) architecture integrated with a K-Nearest Neighbors (KNN) approach to enhance intrusion detection. By first classifying known attack types using CNN and subsequently predicting outliers in new traffic through KNN, RNIDS achieves high accuracy in distinguishing malicious from benign traffic. Our findings demonstrate that the CNN-based model classifies attacks with an accuracy of 98.3% while requiring only 70,252 training parameters, ensuring a resource-efficient approach. However, limitations include potential challenges in detecting highly sophisticated or adaptive attacks, as the model relies on a predefined set of attack types for training.

Yan et al. [34] introduces a few-shot intrusion detection model for IoT, utilizing three primary sub-modules: data augmentation, type conversion, and image categorization to enhance detection accuracy. The model's design aims to address the complexities of identifying low-frequency, sudden network attacks within IoT environments, a challenge often characterized by sparse labeled data. Using two benchmark datasets, CICIDS2018 and N-BaIoT, the model's performance was evaluated and showed measurable improvements over baseline metrics such as accuracy, precision, recall, and F1-Score. Despite these advancements, the model's reliance on data augmentation and type conversion may limit its scalability to new, unseen attack patterns or highly dynamic IoT environments, where the diversity of device behaviors can reduce the effectiveness of predefined augmentation strategies. Additionally, the model's dependence on labeled data could restrict its applicability in real-time environments where labeled samples are sparse or unavailable.

IV. PROPOSED METHODOLOGY

In this section, we have discussed the development of our NIDS which can successfully detect Zero-day attacks. In FIGURE 4, proposed methodology is illustrated.

A. DATA COLLECTION

To develop our DRL agent which can detect zero day attacks we have used networkflow datasets maintained by University Queensland, Australia [8]. We have used NF-BoT-IoT as primary dataset for primary training. For multi dataset validation, we have used NF-ToN-IoT, NF-UNSW-NB15, NF-UQ-NIDS [35] and NF-UNSW-NB15-v2 [36]. Each dataset contains a mix of benign and attack flows; the benign instances are labelled 0, while all kinds of attack types are labelled as 1 similar to fig 2. NF-BoT-IoT is the primary dataset; for the multiple datasets validation, additional datasets were used. NF-BoT-IoT contains 600,100 flows, out of which 586,241 or 97.69% are attacks-Reconnaissance, DDoS, DoS, Theft-and 13,859 or 2.31% are benign. NF-ToN-IoT contains 1,379,274 flows in total, of which there are 1,108,995 attack samples, accounting for 80.4%: Backdoor, DoS, DDoS, Injection, MITM, Password, Ransomware, Scanning, XSS. There are 270,279 benign samples, accounting for 19.6% of the total. In NF-UNSW-NB15, there are 1,623,118 flows; and among them, there are 72,406 attack samples, accounting for 4.46%, including Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms, while 1,550,712 are identified as normal, accounting for 95.54% of the total. In sum, the NF-UQ-NIDS dataset contains 11,994,893 records, out of which 9,208,048-76.77% are benign flows, and 2,786,845-23.23% are labeled as attacks: DDoS, Reconnaissance, Injection, DoS, Brute Force, Password, XSS, Infiltration, Exploits, Scanning, Fuzzers, Backdoor, Bot, Generic, Analysis, Theft, Shellcode, MITM, Worms, Ransomware. The NF-UNSW-NB15-v2 dataset represents an extension of the UNSW-NB15 dataset using the NetFlow technique. It offers enriched features on NetFlow and is labeled based on specific attack categories. This extended dataset, NF-UNSW-NB15-v2, was composed of 2,390,275 data flows, of which 95,053 (3.98%) were labeled as attack samples, and 2,295,222 (96.02%) were labeled as benign. The attack instances are divided into nine different subcategories, which are represented in the flow distribution table for the NF-UNSW-NB15-v2 dataset.

As mentioned previously in Section I, certain attack categories are kept exclusively in the test set and excluded

TABLE 1. Zero day detection training data split strategy.

Dataset	Types of Attack Data Present in Training Data	Types of Attack Data Excluded from Training Data
NF-BoT-IoT	Benign, Reconnaissance and DDoS	DoS and Theft
NF-ToN-IoT	Benign, Backdoor, DDoS, Injection, MITM, Ransomware, Scanning and XSS	DoS and Password
NF-UNSW-NB15	Benign, Fuzzers, Analysis, Exploits, Generic, Reconnaissance, Shellcode and Worms	DoS and Backdoor.
NF-UNSW-NB15-v2	Benign, Fuzzers, Analysis, DoS, Exploits, Shellcode and Worms	Backdoor, Generic and Reconnaissance.
NF-UQ-NIDS	Benign, DDoS, Reconnaissance, Injection, DoS, Password, XSS, Infiltration, Exploits, Scanning, Fuzzers, Bot, Generic, Analysis, MITM and Worms	Shellcode, Brute Force, Theft, Ransomware and Backdoor.

from the training set to enable zero-day attack detection. Table 1 provides an overview of the composition of our training and test sets across different datasets, as we incorporate multiple datasets in our study.

B. DATA PREPROCESSING

The data preprocessing and training setup shown in Algorithm 1 for zero-day attack detection in NIDS begins with loading and cleaning the dataset. Using pandas, the data is loaded from a CSV file, and irrelevant columns such as IPV4_SRC_ADDR and IPV4_DST_ADDR are dropped to focus on relevant features. Label encoding is applied to categorical data (e.g., attack labels), transforming the Label column into a numerical format suitable for agent input. Missing values are then filled with zeros to ensure consistency across the dataset. A critical aspect of this setup is separating data into a training set and a zero-day (test) set. The test set is created by extracting specific attack types (like DoS and Theft) that are unseen in the training data. This setup allows for zero-day detection by training the agent without exposure to these attack types, testing its ability to generalize to previously unseen threats.

For effective agent training, data balancing is handled using KMeans-SMOTE, a synthetic oversampling technique. The training data, which may suffer from class imbalance due to differing attack frequencies, is first scaled using a StandardScaler, followed by K-Means clustering (K=5, random state=42) to create synthetic samples in underrepresented classes. This step reduces bias towards overrepresented classes, making the agent more resilient in identifying rare attacks. After balancing, the data is split into training and testing subsets. For further evaluation, test data is appended with zero-day attack examples from the DoS and Theft categories. The entire preprocessed dataset is then converted into TensorFlow datasets. This step enables efficient data shuffling and batching, facilitating faster training and better agent generalization. This setup ensures that the agent has a well-balanced training dataset, is tested on unseen attacks, and uses structured data input, improving its capability for zero-day attack detection across diverse threat scenarios. By isolating and including zero-day attacks in testing, this approach effectively gauges the agent's ability to detect unknown attack patterns.

1) RATIONALE OF CONSIDERING ALL THE FEATURES OF DATASET INSTEAD OF FEATURE SELECTION

In our study, we intentionally considered all available features from the datasets without performing explicit feature selection. This decision was made to ensure that our DRL-based NIDS learns directly from the full set of network traffic characteristics, allowing the model to automatically identify relevant patterns and dependencies rather than relying on preselected features.

While traditional feature selection methods can help reduce dimensionality, they may also risk omitting subtle but crucial features that contribute to zero-day attack detection. Given that zero-day threats involve previously unseen attack patterns, we aimed to retain as much information as possible to enhance the generalization capabilities of our DRL agent. By feeding the model the complete set of network traffic features, we enable it to develop a deeper, more flexible understanding of attack patterns, reducing the risk of bias introduced by human-driven feature selection.

Furthermore, the nature of DRL allows the model to autonomously learn feature importance during training. Unlike conventional machine learning models that often require manual feature engineering, DRL continuously updates its policy based on observed rewards, dynamically adjusting its attention to the most relevant features over time. This approach enables the model to extract meaningful patterns from raw data without the need for predefined feature selection criteria.

Additionally, network intrusion detection is a highly dynamic field where the relevance of features may shift depending on the nature of evolving threats. A feature that appears unimportant in one dataset or attack scenario might become critical in identifying a new type of attack in the future. By leveraging the full feature set, our model maintains adaptability, ensuring that it does not overlook emerging patterns that could be indicative of sophisticated zero-day attacks. This flexibility is particularly important in handling real-world network traffic, where the distribution and significance of features may change over time.

C. PROPOSED DRL FRAMEWORK

The DRL agent is trained to recognize attacks in a network intrusion detection system using the DQN (Deep Q-Learning

Algorithm 1 Data Preprocessing and Training Setup for Zero-Day Attack Detection in NIDS

Preprocess_and_Train Load the dataset from CSV file using pandas
Drop irrelevant columns (e.g., 'IPV4_SRC_ADDR', 'IPV4_DST_ADDR')
Encode categorical 'Label' column using LabelEncoder
Fill missing values with zero

// Separate data into training and zero-day (test) sets
test_data ← FILTER(data, (data['Attack'] == 'Dos') OR (data['Attack'] == 'Theft'))
train_data ← data[NOT((data['Attack'] == 'Dos') OR (data['Attack'] == 'Theft'))]

// Split training data into features and target
X_train ← train_data without columns 'Label' and 'Attack'
y_train ← train_data['Label']

// Split test data into features and target
X_test ← test_data without columns 'Label' and 'Attack'
y_test ← test_data['Label']

// Standardize features
Initialize StandardScaler as *scaler*

X_train ← *scaler*.fit_transform(X_train)
X_test ← *scaler*.transform(X_test)

// Oversample training data using KMeans-SMOTE to handle class imbalance
Define *kmeans_model* with KMeans (n_clusters = 5)
Initialize KMeans-SMOTE as *smote_model* with *kmeans_model*

(features_resampled, labels_resampled) ← *smote_model*.fit_resample(X_train, y_train)

// Train-test split on resampled data
(features_train, features_test, labels_train, labels_test) ← TRAIN_TEST_SPLIT(features_resampled, labels_resampled, test_size = 0.2)

test_data is not empty **// Process Dos and Theft attack data for testing**
features_dos_theft ← Standardize and filter test_data features
labels_dos_theft ← test_data['Label']

Append *features_dos_theft* to features_test
Append *labels_dos_theft* to labels_test

// Convert datasets into TensorFlow datasets
train_dataset ← TF.DATA.Dataset.from_tensor_slices((features_train, labels_train)).shuffle(1024).batch(64)
test_dataset ← TF.DATA.Dataset.from_tensor_slices((features_test, labels_test)).batch(64)

Return *train_dataset, test_dataset*

Network) framework. In DQN, the agent observes the current environment state, decides actions to maximize cumulative rewards, and learns over time through exploration and exploitation. The learning rate controls weight adjustments, while the epsilon parameter, adjusted across episodes, balances exploration (random actions) and exploitation (agent-driven actions). The agent architecture shown in algorithm 2 comprises a sequential network with dense layers, dropout layers for regularization, and three LSTM (Long Short-Term Memory) layers that capture temporal relationships in network behavior. The output is a dense layer with a softmax activation function. Loss is calculated using categorical cross-entropy, with class_weight adjustments for handling class imbalances. Below is a detailed explanation of each component of the architecture along with the relevant formulas.

In developing DRL agents for zero-day attack detection within NIDS, LSTM networks are chosen for their efficiency in handling sequential data, which holds the key to capturing patterns in network traffic. Our choice of stacked LSTMs is motivated by their superior ability to capture long-term dependencies in sequential network traffic data, which is crucial for identifying evolving attack patterns, including zero-day threats. While architectures such as GRUs, CNNs, and transformers have been explored in other contexts, LSTMs remain particularly well-suited for time-series and sequential data analysis due to their gated memory cells, which help in retaining long-range dependencies. In intrusion detection, temporal correlations between network events are critical, and LSTMs excel at learning these dependencies effectively. The stacked LSTM structure further enhances the model's capacity to learn hierarchical representations, improving generalization to unseen attacks.

With LSTM layers, this enables the DRL agent to learn generalized policies that can effectively discriminate between benign and malicious traffic for zero-day detection without prior exposure to such attacks. The architecture of an LSTM allows the agent to capture critical information from past observations that will help in proper context-aware decisions over new traffic states. Each LSTM cell is comprised of core elements: cell state, hidden state, and multiple gates-notably, forget and input gates-interacting with certain equations (1)-(7) to control the flow of information crucial in zero-day detection scenarios.

Forget gate:

$$f_t = \sigma\left(w_{fh} \cdot h_{t-1} + w_{fx} \cdot x_t + b_f\right) \tag{1}$$

Input Gate:

$$i_t = \sigma\left(w_{ih} \cdot h_{t-1} + w_{ix} \cdot x_t + b_i\right) \tag{2}$$

Input Node:

$$g_t = \tanh\left(w_{gh} \cdot h_{t-1} + w_{gx} \cdot x_t + b_g\right) \tag{3}$$

$$c_{ti} = i_t \cdot g_t \tag{4}$$

$$c_t = c_{ti} + c_{tf} \tag{5}$$

Output gate:

$$o_t = \sigma\left(w_{oh} \cdot h_{t-1} + w_{ox} \cdot x_t + b_o\right) \tag{6}$$

$$h_t = \tanh(c_t) \cdot o_t \tag{7}$$

The ability of LSTMs for selective memory enhancement defines their functionality. It starts with the current cell state, c_t, which has to be updated based on various inputs such as the incoming data x_t and the hidden state from the previous time step h_{t-1}, which gives the context from past network traffic. The most important feature of LSTM is the forget gate, which outputs c_{ti} along with a forget gate signal f_t, enabling it to smartly decide whether to keep information from the previous cell state or throw it away. This automatically filters out the irrelevant network traffic data that might not contribute much to the detection of intrusion.

The next cell state c_t is computed by incorporating c_{ti} with f_t. More specifically, LSTM computes an input gate i_t and an input node g_t using the sigmoid and tanh activation functions, respectively. This is a step that tells what new information needs to be added to the cell state; this is an important step that enables the learning and development of the NIDS agent upon the emerging patterns of the network behavior. The output state o_t will get processed based on inputs x_t and h_{t-1}, which will be responsible for extracting the next hidden state h_t. This is obtained by running the tanh function over the cell state update,, then multiplied by the output gate,. All of this constitutes the mechanism that allows the LSTM to refine its memory, with refinement here taken to mean smoothly adjusting the cell state while retaining the context from past network activity. These types of selective updates are critical for enhancing the agent's competence in effectively detecting and responding with high fidelity to any probable intrusions from the dynamic patterns of network traffic. Finally, the agent is finalized by compilation with a categorical cross-entropy loss and optimized using the Adam optimizer, which changes the weights in the direction of minimizing the loss effectively.

The Act function in algorithm 3 implements an epsilon-greedy policy for action selection in a DRL framework, particularly suited for zero-day detection in NIDS. It takes the current state as input, reshapes it to match the agent's required input format, and then decides whether to explore or exploit based on the random threshold compared to the epsilon parameter. Specifically, with a probability of ϵ, the function selects a random action to encourage exploration of the action space, which may include recognizing unseen attack patterns. Conversely, with a probability of $1 - \epsilon$, it predicts action values (Q-values) using the neural network agent and returns the action corresponding to the highest predicted value. This balance between exploration and exploitation is essential for adapting to the dynamic nature of network environments, especially when the agent must generalize to detect new types of attacks it has not encountered during training. The decision-making process can be mathematically expressed as (8):

$$\text{action} = \begin{cases} \text{random}(0, \text{action\_size} - 1) & \text{if random}() \leq \epsilon \\ \arg\max_a Q(s, a) & \text{otherwise} \end{cases} \tag{8}$$

where $Q(s, a)$ represents the predicted action values for the current state s and actions a.

The Replay function in algorithm 4 implements an Experience Replay mechanism essential for Q-learning in deep reinforcement learning (DRL) frameworks, particularly effective in environments like network intrusion detection systems (NIDS). It begins by sampling a random minibatch of experiences from the agent's memory, where each experience comprises the current state, the action taken, the reward received, the next state, and a `done` flag indicating whether

Algorithm 2 DRL Agent Building

Build_agent
Initialize *agent* with a Sequential layer structure

Add Dense layer with 64 units, activation='relu', input shape=*state_size*
Add Dropout layer with rate=0.1

Reshape output to (1, 64) to feed into LSTM layers

Add LSTM layer with 64 units, return_sequences=True
Add another LSTM layer with 64 units, return_sequences=True

Add a third LSTM layer with 64 units, return_sequences=False

Add Dense layer with 32 units, activation='relu', kernel regularizer (l2=0.01)
Add Dropout layer with rate=0.1

Add final Dense layer with 2 units and softmax activation

Compile *agent* with categorical cross-entropy loss, Adam optimizer, and accuracy metrics

Algorithm 3 Action Selection Using Epsilon-Greedy Policy (Act Function)

Reshape *state* to match the *agent* input shape
if random() $\leq$ epsilon **then**
  return random action between 0 and *action_size* - 1
else
  Predict action-values with *agent* on the *state*
  return the index of the action with the maximum predicted value
end if=0

Algorithm 4 Experience Replay for Q-Learning (Replay Function)

Sample a random *minibatch* of size *batch_size* from *memory*
for each experience (*state*, *action*, *reward*, *next_state*, *done*) in *minibatch* **do**
  Reshape *state* and *next_state* for agent prediction
  Set *target* to *reward* if *done* is True
if not done **then**
  Predict Q-values for *next_state*
  Update *target* to reward + $\gamma \cdot$ max(Q-values for *next_state*)
end if
  Predict Q-values for *state* and set the value of *action* to *target*
  Train the *agent* on *state* with *target_f* using *class_weights*
end for
if epsilon > epsilon_min **then**
  Decay *epsilon* by multiplying it with *epsilon_decay*
end if
= 0

the episode has concluded. This approach allows the agent to learn from a diverse set of experiences rather than just the most recent ones, thus enhancing its learning efficiency and stability.

Within the function, each sampled experience is processed to reshape the states to match the agent's expected input format. The target value for the Q-learning update is computed based on whether the episode has ended. If the `done` flag is `True`, the target is set to the immediate reward. If the episode is ongoing, the target is calculated using the Bellman equation, which combines the immediate reward and the discounted maximum future Q-value for the next state.

This calculation can be represented as (9):

$$\text{target} = \text{reward} + \gamma \cdot \max(Q(\text{next\_state})), \tag{9}$$

where γ is the discount factor that prioritizes immediate rewards over future ones, promoting efficient learning of long-term strategies.

After determining the target, the function predicts the Q-values for the current state and updates the specific value for the action taken to the computed target. This step is crucial for training the agent, as it aligns the agent's predicted Q-values with the newly learned targets. Additionally, the training process incorporates class weights, which help to manage any imbalances in the data, ensuring that the agent learns effectively from all types of experiences. Finally, the Replay function includes an epsilon decay mechanism to gradually reduce the exploration rate over time. If the exploration parameter ϵ is greater than a defined minimum threshold (ϵ_{min}), it decays ϵ by multiplying it with a decay factor (ϵ_{decay}). This adjustment encourages the agent to exploit its learned knowledge more as training progresses, balancing exploration and exploitation for optimal learning. Overall, the `Replay` function significantly enhances the agent's capability to generalize and respond to new patterns, which is particularly important in zero-day detection scenarios in NIDS.

We used several key hyperparameters and components that significantly enhance the agent's learning capabilities. The memory is implemented using a deque with a maximum length of 1000, which facilitates Experience Replay by allowing the agent to store past experiences and sample from them during training. This approach enhances learning stability and diversity by enabling the agent to revisit a wide range of past interactions. The discount rate (γ) is set to 0.97, which balances the importance of immediate versus future rewards, encouraging the agent to consider long-term outcomes while making decisions. The epsilon parameter, initialized at 1.0, represents the exploration rate, which allows the agent to explore the action space and discover new strategies. To prevent the agent from becoming overly reliant on exploration, we established a minimum epsilon value of 0.05 and a slower epsilon decay rate of 0.996, ensuring that the agent maintains sufficient exploration during training while gradually shifting towards exploitation of its learned knowledge. Lastly, the learning rate of 0.001 controls the magnitude of weight updates during training, striking a balance between converging quickly to an optimal policy and maintaining stability in learning. Together, these elements contribute to the robustness and adaptability of the DRL agent in recognizing and responding to dynamic patterns in network traffic, particularly in scenarios involving zero-day attack detection.

We have used some dense (or fully connected) and dropout layers which were crucial for enhancing feature learning and preventing overfitting. Dense layers fully connect each neuron to all neurons in the preceding layer, helping the network capture nuanced relationships within network traffic data for improved anomaly detection. The dense layer operation is mathematically defined as:

$$y = \sigma(W \cdot x + b) \tag{10}$$

where W represents the weight matrix, x is the input vector, b is the bias term, and σ denotes the activation function (ReLU). This setup allows the DRL model to learn complex patterns within the data.

To mitigate overfitting—a common challenge when dealing with imbalanced and rare intrusion data—dropout layers are used. Dropout randomly deactivates a fraction of neurons during each training iteration, preventing the model from becoming overly reliant on specific neurons and thus enhancing its generalization capabilities. The dropout operation can be defined as:

$$y = \text{dropout}(x, p) \tag{11}$$

where p is the dropout rate, representing the probability that a neuron will be "dropped" or set to zero. By incorporating dense and dropout layers, the DRL-based agent can generalize better to unseen attack types (zero-day scenarios) while minimizing overfitting and achieving robust feature learning across varied network traffic data.

The integration of a DQN with LSTM architecture is pivotal in enhancing the agent's ability to detect zero-day attacks in NIDS. The use of LSTMs allows the agent to effectively process and learn from sequential network traffic data, capturing temporal patterns that are essential for identifying anomalies indicative of potential attacks. This temporal understanding is crucial for recognizing new attack patterns that the agent has not encountered during training, thus facilitating generalization. The epsilon-greedy policy promotes exploration of the action space, enabling the agent to experiment with various responses to unseen threats rather than solely relying on previously learned behaviors. Furthermore, the Experience Replay mechanism allows the agent to revisit and learn from a diverse range of past experiences, which reinforces learning and helps mitigate overfitting to specific attack types. The careful management of exploration through epsilon decay ensures that the agent gradually shifts from exploring to exploiting its learned strategies, thereby increasing its detection accuracy over time. Overall, these combined features empower the DQN agent to adapt and respond dynamically to emerging threats, making it highly effective in zero-day detection scenarios where prior knowledge of specific attacks is unavailable.

V. PERFORMANCE ANALYSIS

In this section, we have discussed the performance of proposed DRL agent, multi dataset validation results, impact of synthetic data generation methods and prior study comparison.

A. PERFORMANCE METRICS

To measure performance of our DRL agent, we have used Total Reward, Average Reward, Discounted Reward,

TABLE 2. Performance on multiple datasets of our proposed DRL agent.

Dataset	Accuracy	Precision	Recall	F1 Score	Convergence Rate	Total Reward	Average Reward	Discounted Reward
NF-BoT-IoT	0.99	0.99	0.99	0.99	0.0400	48	0.96	24.91
NF-UQ-NIDS	0.96	0.97	0.96	0.96	0.0400	46	0.92	24.21
NF-ToN-IoT	0.95	0.95	0.95	0.95	0.0000	50	1.00	26.06
NF-UNSW-NB15	0.97	0.98	0.97	0.97	0.0800	46	0.92	23.97
NF-UNSW-NB15-v2	0.98	0.98	0.98	0.98	0.0800	44	0.88	23.55

TABLE 3. Performance comparison with widely used approach in prior work, Dataset: NF-BoT-IoT.

Model/Agent	Accuracy	Precision	Recall	F1 Score	Convergence Rate	Total Reward	Average Reward	Discounted Reward
GRU	0.98	0.98	0.98	0.98	0.0400	48	0.96	25.04
DNN	0.86	0.89	0.86	0.86	0.0400	34	0.68	21.27
CNN	0.95	0.95	0.95	0.95	0.0800	44	0.92	23.07
Stacked LSTM (our Agent)	0.99	0.99	0.99	0.99	0.0400	48	0.96	24.91

Convergence Rate, confusion matrix, precision, recall, f1-score and accuracy. Below are the descriptions of each metric, along with relevant formulas where applicable, contextualized for our work.

The total reward is the cumulative sum of rewards collected by the agent during an episode, providing a general measure of the agent's overall performance. High total rewards indicate successful detection of network intrusions and accurate responses to benign traffic. Each reward r_t received at time t is accumulated over an episode.

$$\text{Total Reward} = \sum_{t=0}^{T} r_t \tag{12}$$

where T is the total number of time steps in an episode.

The average reward measures the agent's average performance per action or per episode, offering insight into how consistently it performs. A high average reward across episodes implies stable and reliable intrusion detection.

$$\text{Average Reward} = \frac{1}{T} \sum_{t=0}^{T} r_t \tag{13}$$

The discounted reward helps the agent focus on immediate versus future rewards, guiding it towards strategies that yield both immediate and long-term benefits in identifying attacks. This measure is calculated by applying a discount factor γ (set at 0.97 in our work) to future rewards, emphasizing earlier rewards over later ones.

$$\text{Discounted Reward} = \sum_{t=0}^{T} \gamma^t \cdot r_t \tag{14}$$

where $\gamma \in [0, 1]$ is the discount factor.

The convergence rate indicates how quickly the DRL agent learns an optimal or near-optimal policy for attack detection. Faster convergence suggests the agent is effectively capturing network traffic patterns. We assess convergence by observing when the agent's average reward plateaus or stabilizes over multiple episodes, ideally converging as ϵ decays.

The confusion matrix provides a breakdown of the agent's predictions versus actual labels, revealing the number of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). This breakdown allows us to understand how well the agent differentiates between benign and malicious traffic.

$$\begin{bmatrix} \text{True Positives (TP)} & \text{False Positives (FP)} \\ \text{False Negatives (FN)} & \text{True Negatives (TN)} \end{bmatrix}$$

Precision measures the agent's ability to correctly identify attacks without mistakenly labeling benign traffic as malicious, which is essential in minimizing false alarms. High precision implies that the agent's positive predictions (intrusions) are mostly accurate.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{15}$$

Recall measures the agent's effectiveness in detecting actual attacks, focusing on minimizing missed detections. High recall indicates that the agent successfully identifies most malicious traffic.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{16}$$

The F1-Score provides a balance between precision and recall, which is especially useful in zero-day scenarios where both false positives and false negatives need to be minimized for effective detection. A higher F1-Score indicates a robust balance between the agent's precision and recall.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{17}$$

Accuracy provides an overall measure of the agent's correct classifications, combining true positives and true negatives over the total samples. While useful, accuracy alone

may not fully capture performance in imbalanced scenarios, so we interpret it alongside other metrics like F1-Score.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{18}$$

These performance metrics together offer a comprehensive view of our DRL agent's capability to detect previously unseen attacks, making it possible to evaluate its accuracy, reliability, and generalization in a zero-day detection context.

B. PERFORMANCE REPORT

We have obtained notable result on detecting seen and unseen attack (not introduced in training set see reference table 1 to understand training and testing data split strategy). All the results are shown in table 4, 5 and 6. The convergence rate of 0.0400 indicates that the agent quickly stabilizes to an effective policy, suggesting it efficiently learns the required patterns for differentiating malicious from benign traffic. The total reward of 48 and a high average reward of 0.96 reflect strong performance across episodes, meaning the agent consistently identifies network behaviors accurately. The discounted reward of 24.91 demonstrates that the agent balances immediate rewards with long-term benefits, focusing on identifying immediate threats while adapting its policy for future network states. Overall, these metrics confirm that the DRL agent is effectively learning and applying its knowledge to reliably detect new attacks in real-time network environments. The total reward, average reward, and discounted reward provide insight into the agent's learning success and reward accumulation. NF-ToN-IoT records the highest total (50) and average (1.00) rewards, indicating a strong learning performance. Discounted rewards, which prioritize immediate over delayed rewards, are highest for NF-ToN-IoT and NF-BoT-IoT, reflecting the agent's ability to prioritize short-term rewards in these datasets while learning a balance between immediate detection and long-term effectiveness in identifying attack patterns.

TABLE 4. Performance on train set, Dataset:NF-BoT-IoT.

Class	Precision	Recall	F1 Score	Support
Benign	1.00	0.98	0.99	467434
Attack	0.98	1.00	0.99	467498
ACCURACY			**0.99**	**934932**
MACRO AVG	**0.99**	**0.99**	**0.99**	**934932**
WEIGHTED AVG	**0.99**	**0.99**	**0.99**	**934932**

TABLE 5. Performance on test set, Dataset:NF-BoT-IoT.

Class	Precision	Recall	F1 Score	Support
Benign	1.00	0.98	0.99	116899
Attack	0.98	1.00	0.99	118743
ACCURACY			**0.99**	**235642**
MACRO AVG	**0.99**	**0.99**	**0.99**	**235642**
WEIGHTED AVG	**0.99**	**0.99**	**0.99**	**235642**

TABLE 6. RL based performance metrics, Dataset:NF-BoT-IoT.

Metric	Score
Convergence Rate	0.04
Total Reward	48
Average Reward	0.96
Discounted Reward	24.91

TABLE 7. Performance metrics of different synthetic data methods, Dataset: NF-BoT-IoT.

Method	Accuracy	Precision	Recall	F1 Score	AUC-ROC
Kmeans SMOTE	0.9895	0.9897	0.9895	0.9895	0.9895
ADYSN	0.9400	0.9410	0.9400	0.9399	0.9399
Borderline-SMOTE	0.9376	0.9397	0.9376	0.9375	0.9376
SMOTE	0.8533	0.8629	0.8533	0.8522	0.8525

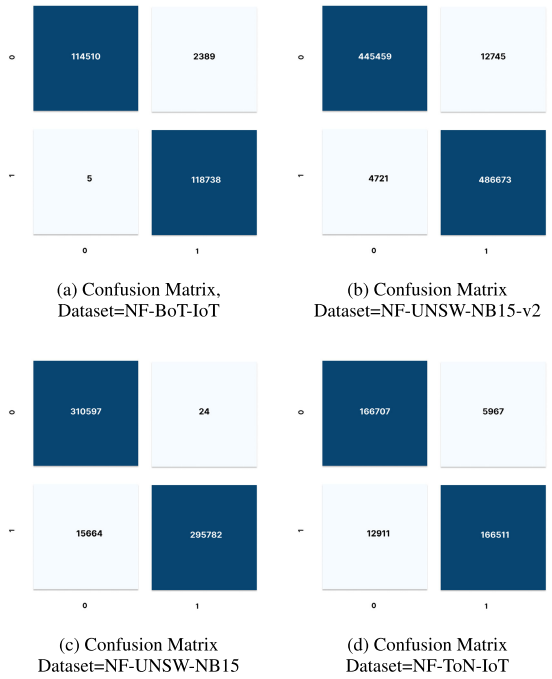
C. MULTI DATASET VALIDATION

Getting high performance in a single dataset is not enough to claim robustness. Need of multiple dataset validation is required to test DRL agent's true performance. We have conducted multi dataset validation strategy for testing our DRL agent's adaptive features shown in table 2. The train and test split was followed according to table 1. Each datasets are distinct in volume, size and attack types. Agent shows high accuracy across all datasets, with values ranging from 0.95 to 0.99. The highest accuracy (0.99) is achieved on the NF-BoT-IoT dataset, indicating reliable classification of network traffic in this dataset. The convergence rate, which measures the speed at which the agent learns an optimal policy, varies between datasets. Datasets NF-UNSW-NB15 and NF-UNSW-NB15-v2 have a higher convergence rate (0.0800), indicating that the agent learns more rapidly on these datasets. Meanwhile, NF-ToN-IoT has a convergence rate of 0.0000, suggesting the agent required fewer episodes to stabilize, potentially indicating simpler patterns or reduced complexity in this dataset. Overall, the DRL agent demonstrates high accuracy, precision, and recall across all datasets, with variations in convergence rate and reward metrics reflecting the differing levels of complexity in network patterns for each dataset. These results affirm the agent's adaptability and efficiency in zero-day attack detection for various types of network traffic. We have also conducted prior study comparison in table 3.

D. IMPACT OF SYNTHETIC DATA GENERATION METHOD

Network flow data exhibit significant class imbalance, with intrusion events occurring less frequently than benign activities. To address this issue, we employed various data balancing techniques and found that k-means SMOTE outperformed other methods such as traditional SMOTE, Borderline-SMOTE and ADASYN shown in table 7.

Table 7 showcases the performance metrics of various synthetic data balancing methods applied to the NF-BoT-IoT dataset, focusing on their effectiveness in improving classification outcomes for network flow data. K-means SMOTE outperformed all other techniques,



(a) Confusion Matrix, Dataset=NF-BoT-IoT

(b) Confusion Matrix Dataset=NF-UNSW-NB15-v2

(c) Confusion Matrix Dataset=NF-UNSW-NB15

(d) Confusion Matrix Dataset=NF-ToN-IoT

FIGURE 5. Confusion matrix in different network dataset.

achieving an accuracy of 0.9895, along with high precision (0.9897) and recall (0.9895), resulting in a robust F1 score of 0.9895 and an AUC-ROC of 0.9895, indicating excellent classification capabilities. In contrast, ADASYN and Borderline-SMOTE demonstrated lower performance, with accuracies of 0.9400 and 0.9376, respectively, showing a good but diminished ability to identify true positives. Traditional SMOTE yielded the lowest performance, with an accuracy of only 0.8533 and significantly lower precision and recall metrics, reflecting its ineffectiveness in distinguishing between benign and intrusion events.

E. UNDERSTANDING FEATURE CONTRIBUTION

LIME, which stands for Local Interpretable Model-Agnostic Explanations, is a technique used to interpret complex models. It does this by creating an interpretable model that approximates the decision boundary around the input instance, and then it takes that same instance but changes values of its features up/down by adding perturbative noises and sees what decisions have changed. LIME provides the weights about the contribution of each feature in the prediction, in the localized scope, by analyzing the effect of each feature in this localized scope. This kind of interpretation gives the user an insight into which features are more responsible for a particular classification; these decisions in models could well be difficult to interpret, like neural networks and ensemble models.

1) SENARIO: 1: ATTACK

The LIME output in fig. 6 shows that the DRL agent classified this instance as an "Attack" with 100% confidence, displaying the most influential features contributing to this decision. On the left, features like IN_PKTS (> -0.04), IN_BYTES (> -0.04), and L4_SRC_PORT ($<= -0.61$) indicate a tendency towards "Benign," but these weren't strong enough to change the classification. On the right, features such as OUT_BYTES ($<= -0.03$) and OUT_PKTS ($<= -0.03$) slightly favor "Attack," and the combined effect of these smaller weights collectively supports the "Attack" classification. This output illustrates how specific feature values influenced the model's decision, providing insights into the key drivers of the DRL agent's high-confidence prediction.

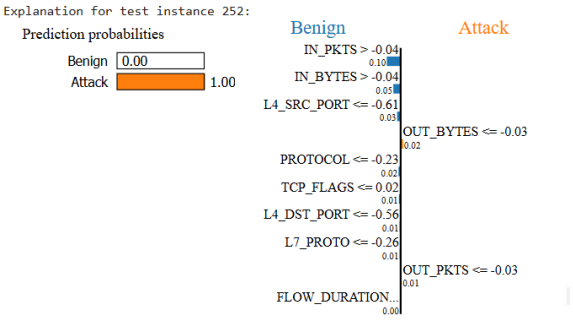


FIGURE 6. Lime output for label: Attack.

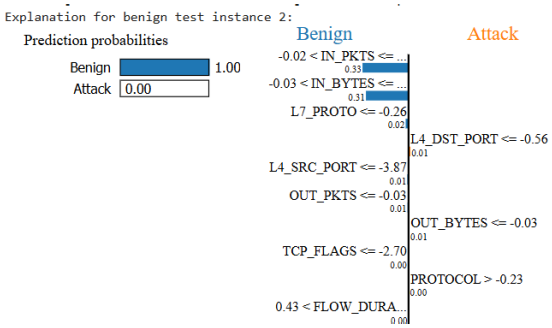


FIGURE 7. Lime output for label: Benign.

2) SENARIO: 2: NON-ATTACK

This LIME output in fig. 7 shows that the DRL agent classified this instance as "Benign" with 100% confidence. The explanation reveals that features such as IN_PKTS ($-0.02 <$ IN_PKTS $<= 0.33$), IN_BYTES ($-0.03 <$ IN_BYTES $<= 0.31$), and L7_PROTO ($<= -0.26$) have the largest weights contributing to the "Benign" classification, pushing the prediction confidently toward non-malicious traffic. Smaller contributions towards "Attack" come from features like L4_DST_PORT ($<= -0.56$) and OUT_BYTES ($<= -0.03$), but these were outweighed by the factors indicating benign behavior. This interpretation highlights the specific feature ranges that led the model to classify the instance as benign, providing insights into the primary factors behind the decision.

VI. DISCUSSION

The implementation of a stacked LSTM architecture has proven to be a pivotal factor in improving the learning

capabilities of our DRL agent. LSTMs are particularly well-suited for sequence prediction tasks due to their ability to retain information over long periods, which is essential for analyzing network traffic patterns. This characteristic allows the model to effectively learn from historical data and make informed predictions about potential intrusions, even when faced with novel attack vectors. The results indicate that our model not only achieves high accuracy but also maintains robustness across various datasets, underscoring the effectiveness of the LSTM architecture in this domain.

The unique zero-day training strategy employed in our research is a significant innovation. By reserving certain attack categories, such as Denial of Service (DoS) and backdoor attacks, for testing, we simulate real-world scenarios where an NIDS may encounter previously unknown threats. The ability of our DRL agent to successfully detect these zero-day attacks, despite not being trained on them, demonstrates the model's adaptability and generalization capabilities. This aspect is particularly crucial in the ever-evolving landscape of cybersecurity, where new attack vectors emerge frequently.

The robustness of our proposed DRL agent is further validated through multi-dataset testing. By evaluating the model's performance across different datasets, we establish its effectiveness in detecting both known and unknown attacks. This multi-faceted validation approach is essential for demonstrating the practical applicability of our solution in real-world scenarios, where network environments can vary significantly. The high accuracy and F1 scores achieved across these datasets reinforce the reliability of our model.

A. JUSTIFICATION FOR NOT USING CROSS-VALIDATION

We would like to clarify that our model has been rigorously evaluated on multiple datasets, which serves as a strong validation strategy for assessing its generalizability and robustness. Instead of using k-fold cross-validation on a single dataset, we evaluated the model across multiple independent datasets, each containing diverse network traffic patterns and attack scenarios. This approach ensures that our model is not overfitting to a particular dataset and provides a more realistic assessment of its performance in different network environments. Our model employs an LSTM-based deep reinforcement learning framework, which relies on sequential patterns in network traffic data. Traditional k-fold cross-validation randomly partitions the dataset, which can disrupt the temporal dependencies that are crucial for learning attack patterns in network traffic. Instead, we maintained the chronological order of the data, ensuring that the model learns patterns effectively while preventing data leakage. Since our primary goal is to develop a model that can generalize well to new, unseen attack patterns, testing on multiple datasets that simulate different network environments is more aligned with real-world deployment scenarios than k-fold cross-validation. Our results demonstrate consistent performance across these datasets, reinforcing the model's robustness.

B. RATIONALE OF USING SMOTE AND ITS VARIANTS OVER COST SENSITIVE METHODS

Our decision to prioritize oversampling techniques such as SMOTE, Borderline-SMOTE, K-Means SMOTE, and ADASYN was based on the specific challenges of zero-day attack detection in NIDS. Cost-sensitive learning methods are highly effective for handling class imbalance in traditional classification problems by adjusting misclassification penalties. However, zero-day attack detection is fundamentally different because the model must generalize to unseen attack types rather than just rebalance known ones.

Oversampling methods like SMOTE and its variants generate synthetic samples that improve the agent's ability to recognize patterns in minority classes without altering the loss function dynamics. Cost-sensitive methods, in contrast, modify the learning process by assigning different penalties to misclassified samples. While this is useful in some contexts, it may not necessarily improve the ability of a DRL-based agent to generalize to new, unseen attack categories.

Applying cost-sensitive learning in our setting would require predefined cost matrices based on attack severity or misclassification impact. However, in a zero-day detection scenario, the severity of an attack is not always known in advance. Weighting certain attack categories more heavily could bias the model toward known attacks while reducing its ability to generalize to unseen threats.

VII. LIMITATION

The computational resources required to train a deep reinforcement learning model with a stacked LSTM architecture can be substantial, potentially limiting the accessibility of the model for environments with constrained resources.

VIII. FUTURE WORK

For future work, we suggest exploring lightweight model architectures that can achieve similar zero-day detection capabilities with reduced computational demands. Incorporating adversarial learning techniques to improve robustness against evolving attack patterns could further enhance the model's real-world applicability. Additionally, exploring few-shot or meta-learning approaches may help bridge the gap in detecting highly divergent zero-day attacks by allowing the model to generalize more effectively across different attack types.

IX. CONCLUSION

This paper introduces a novel DRL-based NIDS capable of zero-day attack detection by utilizing a stacked LSTM architecture and a targeted zero-day training strategy. By omitting specific attack types in training and evaluating the agent's ability to detect these in testing, we demonstrated that the agent could effectively identify previously unseen attack types, showcasing its robustness in detecting both known and unknown intrusions. Our comparative analysis of various data balancing techniques further highlighted the impact of

handling class imbalance on agent performance, with methods like K-means SMOTE, SMOTE, Borderline-SMOTE, and ADASYN proving valuable in different contexts. The validation across multiple benchmark datasets—NF-BoT-IoT, NF-ToN-IoT, NF-UNSW-NB15, NF-UNSW-NB15-v2, and NF-UQ-NIDS—confirms the agent's potential as an effective tool for real-time network intrusion detection in diverse scenarios. This research contributes to advancing NIDS by providing a promising approach to zero-day detection, paving the way for further innovations that enhance network security in an era of rapidly evolving cyber threats.

REFERENCES

[1] U. Ahmed, Z. Jiangbin, A. Almogren, S. Khan, M. T. Sadiq, A. Altameem, and A. U. Rehman, "Explainable AI-based innovative hybrid ensemble model for intrusion detection," *J. Cloud Comput.*, vol. 13, no. 1, Oct. 2024, Art. no. 150, doi: 10.1186/s13677-024-00712-x.

[2] H. Zeng and H. Chen, "Network intrusion detection based on LSTM," *Frontiers Sci. Eng.*, vol. 4, no. 9, pp. 131–137, Sep. 20, 2024, doi: 10.54691/p4w71z56.

[3] V. Ravi, "Deep learning-based network intrusion detection in smart healthcare enterprise systems," *Multimedia Tools Appl.*, vol. 83, no. 13, pp. 39097–39115, Oct. 7, 2023, doi: 10.1007/s11042-023-17300-x.

[4] Y.-D. Lin, H.-X. Huang, D. Sudyana, and Y.-C. Lai, "AI for AI-based intrusion detection as a service: Reinforcement learning to configure models, tasks, and capacities," *J. Netw. Comput. Appl.*, vol. 229, Sep. 2024, Art. no. 103936, doi: 10.1016/j.jnca.2024.103936.

[5] T. Nandy, R. M. Noor, R. Kolandaisamy, M. Y. I. Idris, and S. Bhattacharyya, "A review of security attacks and intrusion detection in the vehicular networks," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 36, no. 2, Feb. 2024, Art. no. 101945, doi: 10.1016/j.jksuci.2024.101945.

[6] D. Manivannan, "Recent endeavors in machine learning-powered intrusion detection systems for the Internet of Things," *J. Netw. Comput. Appl.*, vol. 229, Sep. 2024, Art. no. 103925, doi: 10.1016/j.jnca.2024.103925.

[7] Z. Ahmad, A. S. Khan, C. W. 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. 16, 2020, Art. no. e4150, doi: 10.1002/ett.4150.

[8] School Inf. Technol. Electr. Eng. *ML-Based NIDS Datasets*. Accessed: Jun. 6, 2024. [Online]. Available: https://staff.itee.uq.edu.au/marius/NIDS_datasets/

[9] Z. Yang, X. Liu, T. Li, D. Wu, J. Wang, Y. Zhao, and H. Han, "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.

[10] M. J. F. Cevallos, A. Rizzardi, S. Sicari, and A. C. Porisini, "NERO: Neural algorithmic reasoning for zero-day attack detection in the IoT: A hybrid approach," *Comput. Secur.*, vol. 142, Jul. 2024, Art. no. 103898, doi: 10.1016/j.cose.2024.103898.

[11] V. G. da Silva Ruffo, D. M. B. Lent, M. Komarchesqui, V. F. Schiavon, M. V. O. de Assis, L. F. Carvalho, and M. L. Proença, "Anomaly and intrusion detection using deep learning for software-defined networks: A survey," *Expert Syst. Appl.*, vol. 256, Dec. 2024, Art. no. 124982, doi: 10.1016/j.eswa.2024.124982.

[12] R. Sultana, J. Grover, and M. Tripathi, "Intelligent defense strategies: Comprehensive attack detection in VANET with deep reinforcement learning," *Pervas. Mobile Comput.*, vol. 103, Oct. 2024, Art. no. 101962, doi: 10.1016/j.pmcj.2024.101962.

[13] T. Ige, C. Kiekintveld, A. Piplai, A. Wagler, O. Kolade, and B. H. Matti, "An in-depth investigation into the performance of state-of-the-art zero-shot, single-shot, and few-shot learning approaches on an out-of-distribution zero-day malware attack detection," in *Proc. Int. Symp. Networks, Comput. Commun. (ISNCC)*, Washington, DC, USA, 2024, pp. 1–6, doi: 10.1109/ISNCC62547.2024.10758952.

[14] D. Di Monda, A. Montieri, V. Persico, P. Voria, M. De Ieso, and A. Pescapè, "Few-shot class-incremental learning for network intrusion detection systems," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 6736–6757, 2024, doi: 10.1109/OJCOMS.2024.3481895.

[15] H. Benaddi, K. Ibrahimi, A. Benslimane, M. Jouhari, and J. Qadir, "Robust enhancement of intrusion detection systems using deep reinforcement learning and stochastic game," *IEEE Trans. Veh. Technol.*, vol. 71, no. 10, pp. 11089–11102, Oct. 2022, doi: 10.1109/TVT.2022.3186834.

[16] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. T. Karim, S. Rashidi, M. Hosseinzadeh, and A. M. Rahmani, "Deep learning-based intrusion detection systems: A systematic review," *IEEE Access*, vol. 9, pp. 101574–101599, 2021, doi: 10.1109/ACCESS.2021.3097247.

[17] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019, doi: 10.1109/COMST.2018.2847722.

[18] M. Al-Fawa'reh, J. Abu-Khalaf, P. Szewczyk, and J. J. Kang, "MalBoT-DRL: Malware botnet detection using deep reinforcement learning in IoT networks," *IEEE Internet Things J.*, vol. 11, no. 6, pp. 9610–9629, Mar. 2024, doi: 10.1109/JIOT.2023.3324053.

[19] M. He, X. Wang, P. Wei, L. Yang, Y. Teng, and R. Lyu, "Reinforcement learning meets network intrusion detection: A transferable and adaptable framework for anomaly behavior identification," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 2, pp. 2477–2492, Apr. 2024, doi: 10.1109/TNSM.2024.3352586.

[20] X. Ma and W. Shi, "AESMOTE: Adversarial reinforcement learning with SMOTE for anomaly detection," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 943–956, Apr. 2021, doi: 10.1109/TNSE.2020.3004312.

[21] M. Wang, N. Yang, N. J. Forcade-Perkins, and N. Weng, "ProGen: Projection-based adversarial attack generation against network intrusion detection," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 5476–5491, 2024, doi: 10.1109/TIFS.2024.3402155.

[22] C. Park, J. Lee, Y. Kim, J.-G. Park, H. Kim, and D. Hong, "An enhanced AI-based network intrusion detection system using generative adversarial networks," *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2330–2345, Feb. 2023, doi: 10.1109/JIOT.2022.3211346.

[23] R. Soleymanzadeh and R. Kashef, "Efficient intrusion detection using multi-player generative adversarial networks (GANs): An ensemble-based deep learning architecture," *Neural Comput. Appl.*, vol. 35, no. 17, pp. 12545–12563, Mar. 2023, doi: 10.1007/s00521-023-08398-z.

[24] R. Tahri, Y. Balouki, A. Jarrar, and A. Lasbahani, "Intrusion detection system using machine learning algorithms," *ITM Web Conf.*, vol. 46, May 2022, Art. no. 02003, doi: 10.1051/itmconf/20224602003.

[25] H. Wang, Y. Wang, and Y. Guo, "A novel approach of unknown network attack detection based on zero-shot learning," in *Proc. IEEE Int. Conf. Data Sci. Comput. Appl. (ICDSCA)*, Dalian, China, Oct. 2021, pp. 312–318, doi: 10.1109/ICDSCA53499.2021.9650182.

[26] B. Xu, B. Wang, X. Chen, J. Zhao, and G. He, "A CGAN-based few-shot method for zero-day attack detection in the Internet of Vehicles," in *Proc. 11th Int. Conf. Adv. Cloud Big Data (CBD)*, Danzhou, China, Dec. 2023, pp. 98–103, doi: 10.1109/cbd63341.2023.00026.

[27] A. Arun, A. S. Nair, and A. G. Sreedevi, "Zero day attack detection and simulation through deep learning techniques," in *Proc. 14th Int. Conf. Cloud Comput., Data Sci. Eng.*, Noida, India, Jan. 2024, pp. 852–857, doi: 10.1109/confluence60223.2024.10463429.

[28] C. D. Luu, V. Q. Nguyen, T. S. Pham, and N.-A. Le-Khac, "A zero-shot deep learning approach for unknown IoT Botnet attack detection," in *Proc. RIVF Int. Conf. Comput. Commun. Technol. (RIVF)*, Hanoi, Vietnam, Dec. 2023, pp. 278–283, doi: 10.1109/rivf60135.2023.10471793.

[29] I. S. Bangroo and R. Kumar, "Zérosdetect: Phishing URL detection with quantum-driven zero-shot learning," in *Proc. 9th Int. Conf. Signal Process. Commun. (ICSC)*, Noida, India, Dec. 2023, pp. 498–503, doi: 10.1109/icsc60394.2023.10440725.

[30] V.-T. Hoang, Y. A. Ergu, V.-L. Nguyen, and R.-G. Chang, "Security risks and countermeasures of adversarial attacks on AI-driven applications in 6G networks: A survey," *J. Netw. Comput. Appl.*, vol. 232, Dec. 2024, Art. no. 104031, doi: 10.1016/j.jnca.2024.104031.

[31] E. D. O. Andrade, J. Guérin, J. Viterbo, and I. G. B. Sampaio, "Adversarial attacks and defenses in person search: A systematic mapping study and taxonomy," *Image Vis. Comput.*, vol. 148, Aug. 2024, Art. no. 105096, doi: 10.1016/j.imavis.2024.105096.

[32] M. Li, N. Mour, and L. Smith, "Machine learning based on reinforcement learning for smart grids: Predictive analytics in renewable energy management," *Sustain. Cities Soc.*, vol. 109, Aug. 2024, Art. no. 105510, doi: 10.1016/j.scs.2024.105510.

[33] R. Desai and T. G. Venkatesh, "Robust network intrusion detection systems for outlier detection," in *Proc. IEEE 27th Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Paris, France, Nov. 2022, pp. 140–146, doi: 10.1109/CAMAD55695.2022.9966883.

[34] Y. Yan, Y. Yang, Y. Gu, and F. Shen, "A few-shot intrusion detection model for the Internet of Things," in *Proc. 3rd Int. Conf. Electron. Inf. Eng. Comput. Sci. (EIECS)*, Changchun, China, Sep. 2023, pp. 531–537, doi: 10.1109/EIECS59936.2023.10435498.

[35] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "NetFlow datasets for machine learning-based network intrusion detection systems," in *Proc. Int. Conf. Big Data Technol. Appl.* Cham, Switzerland: Springer, 2021, pp. 117–135, doi: 10.1007/978-3-030-72802-1.

[36] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a standard feature set for network intrusion detection system datasets," *Mobile Netw. Appl.*, vol. 27, no. 1, pp. 357–370, Nov. 2021, doi: 10.1007/s11036-021-01843-0.

KHORSHED ALAM received the Master of Science degree in computer science and engineering (CSE) from United International University (UIU). He is currently a Research Assistant with Independent University at Bangladesh (IUB) and Southeast University (SEU). With two years of professional experience as a Software Engineer specializing in android app development. He has published high-ranked research articles in the domains of AI, biomedical engineering, signal processing, and cyber security, collaborating with esteemed professors and Ph.D. scholars from institutions, such as the University of Houston, Virginia Tech, University of Southern Denmark, Laval University, UIU, IUB, and SEU. He has demonstrated academic excellence, earning several prestigious accolades, including the Emerging Leader Award 2024, the Dean's List Award, multiple research grants, and Merit Scholarships. He serves as a Journal Article Reviewer for IEEE Access (Q1, Computer Science). He also reviews for Springer Nature's *The Journal of Supercomputing* (Q2), Vehicular Technology Conference, ISBCom, TEMSCON ASPAC, and ICMI. His long-term goal is to contribute to international cybersecurity policy and defense initiatives, advancing the missions of SDG 9 and SDG 16 through technological innovation and responsible AI.

MD FAHAD MONIR (Member, IEEE) received the B.Eng. degree (Hons.) in communication engineering from International Islamic University Malaysia (IIUM), Malaysia, in 2014, the M.Sc. degree in information technology from the KTH Royal Institute of Technology, Sweden, in 2016, and the M.Eng. degree in communication engineering from the University of Trento, Italy, in 2017. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Virginia Tech, USA. From 2018 to 2019, he was a full-time Lecturer with American International University of Bangladesh. Since 2019, he has been a Senior Lecturer with the Department of Computer Science and Engineering, Independent University at Bangladesh. His current research interests include software-defined networking.

MD JUNAYED HOSSAIN received the B.Sc. degree in CSE from Independent University at Bangladesh (IUB). He is currently an Adjunct Lecturer of computer science and engineering with IUB. Previously, he was a Graduate Research Engineer, he contributed to a project on stroke biomarker identification via biomedical signal processing. His research interests include biomedical signal processing, machine learning, computer vision, and NLP, with IEEE publications on topics, such as Bangla speech emotion recognition and satellite image classification. Currently, he is exploring research in intrusion detection and medical image processing and is actively seeking Ph.D. opportunities.

MOHAMMAD SHORIF UDDIN (Senior Member, IEEE) received the Bachelor of Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology in 1991, the Master of Technology Education from Shiga University, Japan, in 1999, the Doctor of Engineering (Ph.D.) degree from Kyoto Institute of Technology, Japan, in 2002, and the M.B.A. degree from Jahangirnagar University in 2013. His academic journey began in 1991 as a Lecturer at Bangladesh Institute of Technology, Chittagong (now CUET). In 1992, he joined the Department of Computer Science and Engineering at Jahangirnagar University, where he currently serves as a full-time Professor. Throughout his career, he has made significant contributions, including serving as Chairperson of the Computer Science and Engineering Department and as Teacher-in-Charge of the ICT Cell at Jahangirnagar University. He has an extensive international research background, having conducted postdoctoral research at several renowned institutions, including the Bioinformatics Institute in Singapore, Toyota Technological Institute in Japan, Kyoto Institute of Technology in Japan, Chiba University in Japan, Saitama University in Japan, Bonn University in Germany, and the Institute of Automation, Chinese Academy of Sciences in China. His professional and research interests encompass a wide range of areas, including computer vision, image security, data science, artificial intelligence, and machine learning. He has a prolific publication record, with over 250 research papers published in prestigious venues such as Elsevier, Springer, IEEE, ACM, Wiley, Oxford University Press, SPIE, and Optica, accumulating around 5000 Google Scholar citations. Additionally, he holds two patents for his innovative scientific work and has edited numerous books and authored several book chapters, published by Springer and Elsevier. Notably, in 2004, he gained worldwide recognition from prominent news and television media outlets, including CNN, BBC, Reuters, The New York Times, The Japan Times, and Associated Press, for his groundbreaking invention of a visually impaired navigational aid called the "Electronic Eye." Furthermore, he mentored and coached the Jahangirnagar University ACM ICPC World Finals Teams in 2015 and 2017 and supervised numerous doctoral and Master's theses. He has been serving as the Vice Chancellor of Green University of Bangladesh since May 16, 2024. He is honored as a Fellow of IEB and BCS, and serves as an Associate Editor of IEEE Access.

MD. TAREK HABIB received the Bachelor of Science degree in computer science from BRAC University, in 2006, the M.S. degree in computer science and engineering from North South University, in 2009, and the Ph.D. degree from the Department of Computer Science and Engineering, Jahangirnagar University, in 2022. At Independent University at Bangladesh, he holds the position of an Assistant Professor with the Department of Computer Science and Engineering. He previously held the position of an Associate Professor with the Department of Computer Science and Engineering, Daffodil International University. He loves doing research. He has numerous publications published in international journals and conference proceedings. Artificial intelligence, particularly machine learning, artificial neural networks, computer vision, and natural language processing, is his area of interest. His research interests include not only AI but also the IoT and mathematics.