applied sciences

MDPI

Article

A Reinforcement Learning Framework with Oversampling and Undersampling Algorithms for Intrusion Detection System

Najmeh Abedzadeh * and Matthew Jacobs

Computer Science, Catholic University of America, Washington, DC 20017, USA; jacobsmj@cua.edu
* Correspondence: abedzadeh@cua.edu

Abstract: Intrusion detection systems (IDSs) play a pivotal role in safeguarding networks and systems against malicious activities. However, the challenge of imbalanced datasets significantly impacts IDS research, skewing learning models towards the majority class and diminishing accuracy for the minority class. This study introduces the Reinforcement Learning (RL) Framework with Oversampling and Undersampling Algorithm (RLFOUA) to address imbalanced datasets. RLFOUA combines RL with diverse resampling algorithms, creating an adaptive learning environment. It integrates the novel True False Rate Synthetic Minority Oversampling Technique (TFRSMOTE) algorithm, emphasizing data-level approaches. Additionally, RLFOUA employs a cost-sensitive approach based on classification metrics. Using the CSE-CIC-IDS2018 and NSL-KDD datasets, RLFOUA demonstrates substantial improvement over existing resampling techniques. Achieving an accuracy of 0.9981 for NSL-KDD and 0.9846 for CSE-CIC-IDS2018, the framework's performance is evaluated using F1 score, accuracy, precision, recall, and a proposed Index Metric (IM). RLFOUA presents a significant advancement in addressing class imbalance challenges in IDS. It shows an average accuracy improvement of 21.5% compared to the recent resampling technique AESMOTE on the NSL-KDD dataset.

Keywords: imbalance; intrusion detection system; machine learning; reinforcement learning; resampling algorithms

check for updates

Citation: Abedzadeh, N.; Jacobs, M. A Reinforcement Learning Framework with Oversampling and Undersampling Algorithms for Intrusion Detection System. *Appl. Sci.* **2023**, *13*, 11275. https://doi.org/10.3390/app132011275

Academic Editor: Luis Javier Garcia Villalba

Received: 9 September 2023
Revised: 7 October 2023
Accepted: 12 October 2023
Published: 13 October 2023

1. Introduction

Cyber-attacks are on the rise due to a combination of evolving tactics by malicious actors and an expanding digital landscape [1]. With the proliferation of connected devices and the increasing dependence on online platforms, there are more opportunities for cybercriminals to exploit vulnerabilities. Additionally, the sophistication of attacks has grown, making them harder to detect and defend against. To defend against these threats, intrusion detection systems (IDSs) must continuously improve their ability to detect new types of attacks [2]. However, a major challenge for machine-learning-based IDSs is that the data used for training and testing are often imbalanced. This means there are usually more normal activities than intrusion instances, making the learning models lean towards recognizing regular behaviors. This can result in cyber-attacks going unnoticed or being misclassified.

Dealing with imbalanced datasets has been a significant focus for researchers, especially in security applications where attack data are limited [2]. This challenge becomes even more apparent when trying to classify different types of attacks, as some categories may have very few examples or even just one. To tackle this challenge, our research introduces an innovative algorithm to improve training with imbalanced datasets. Our approach includes three main novel parts: The Reinforcement Learning (RL) Framework with Oversampling and Undersampling Algorithms (RLFOUA), the True False Rate Synthetic Minority Oversampling Technique (TFRSMOTE) resampling algorithm, and a new evaluation metric called the Index Metric (IM).

In the context of intrusion detection systems (IDSs), a system's response to a detected potential intrusion can be categorized as either passive or active. A passive response involves the IDS solely in the process of detecting and notifying about suspicious activities without taking direct action against them. In this study, our focus is on the development and evaluation of resampling techniques using RL algorithms to enhance the performance of IDS in detecting attacks. Our approach solely focuses on detection, without any influence on the response, whether passive or active, when a potential intrusion is detected by an IDS. The aim is to improve the learning process and model performance in distinguishing between normal and malicious activities, ultimately enhancing the accuracy of intrusion detection. Therefore, our approach is centered around refining the data preprocessing step to provide more effective training data for the machine learning algorithms used in the detection process. Our work does not affect whether or not the IDS takes active response, such as blocking or mitigating detected intrusions. Our primary objective lies in improving the detection accuracy. RL offers a promising approach for intrusion detection due to its ability to adapt and learn from dynamic environments. Traditional rule-based systems or static models may struggle to keep pace with the evolving tactics of cyber-attackers. RL, on the other hand, enables the IDS to continuously refine its strategies based on feedback from its environment. This adaptability is crucial in the ever-changing landscape of cyber threats.

In this paper, we introduce the RLFOUA framework, which is designed to balance datasets and train machine learning models for better performance. RLFOUA uses a special learning environment guided by RL principles, which learns from feedback provided by our new evaluation metric. An important component of RLFOUA is the TFRSMOTE resampling algorithm, which systematically generates synthetic data. To measure the effectiveness of our results, we use various metrics like F1 score, accuracy, recall, and precision. We rigorously test the RLFOUA algorithm using two well-known datasets: CSE-CIC-IDS2018 and NSL-KDD. The former focuses on distinguishing between normal and infiltration samples, while the latter classifies 23 different types of attacks.

In summary, this paper makes significant contributions in the following ways:

- Implementing the novel RLFOUA framework, an innovative automatic learning environment that pioneers the use of RL techniques in dataset resampling. This approach dynamically determines the next state based on the current state and past rewards, setting it apart from traditional resampling methods.
- Proposing the innovative TFRSMOTE resampling algorithm, a pioneering solution for handling imbalanced datasets in a dynamic environment. This algorithm generates synthetic data from the minority class for oversampling and strategically reduces the majority class for effective undersampling, marking a departure from conventional resampling techniques.
- Presenting the new IM as an original contribution to the field. This novel metric offers a comprehensive evaluation of algorithm performance, considering a range of critical factors not addressed by existing metrics.

2. Related Works

The problem of imbalanced datasets has long been recognized as a challenging research issue, particularly in the field of security [1]. Numerous studies and surveys have explored the use of machine learning models for IDS dataset classification, with several attempts made to address the imbalanced nature of the datasets through undersampling, oversampling, and combination techniques [2]. Oversampling techniques aim to increase the representation of minority classes, while undersampling techniques aim to reduce the size of the majority class. More recently, RL has been investigated in combination with other sampling models for handling imbalanced datasets [3]. In this section, we will review the contemporary advances in resampling algorithms and RL concerning their application to imbalanced datasets.

2.1. Resampling Algorithms

Phetlasy et al. [4] presented a novel approach to improving the performance of IDSs by combining SMOTE with a sequential classifier method. SMOTE, a prevalent oversampling technique used to address imbalanced datasets, alleviates the scarcity of samples in minority classes by generating synthetic instances [5]. This method identifies neighboring examples in the feature space, establishes connections between them, and subsequently introduces new samples along this line. It is crucial to underscore that this process does not involve a literal increase in cyber-attacks; instead, it strategically enriches the dataset with synthetic instances that represent minority class patterns. Through this strategic synthesis of data points within the feature space, SMOTE plays a pivotal role in redressing class imbalances, ultimately enabling the creation of robust machine learning models for attack detection. Phetlasy et al. proposed a methodology that utilizes SMOTE to address class imbalance in the IDS dataset and enhance the accuracy of intrusion detection. By integrating SMOTE with sequential classifiers on the NSL-KDD dataset of Network IDS (NIDS), the study demonstrated improved sensitivity (0.95) and accuracy (0.93) in detecting intrusions.

Qazi et al. [6] conducted a study to examine the impact of feature selection, SMOTE, and undersampling methods on class imbalance classification. The authors explored various techniques to address imbalanced datasets in classification tasks, evaluating their performance on the KDD CUP 99 IDS dataset using metrics such as true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The study provided insights into the effectiveness of feature selection and sampling techniques in enhancing classification accuracy and handling class imbalance across different domains. The incorporation of SMOTE into specific machine learning algorithms resulted in noteworthy enhancements, exemplified by the random forest (RF) exhibiting a true positive rate of 1.0 for normal activities, 0.998 for DOS, 0.937 for probe, 0.975 for R2L, and 0.673 for U2R attacks. However, it is important to note that direct comparisons with our method are not feasible due to the dissimilarities in datasets and the absence of comprehensive classification metrics in their evaluation.

Tesfahun et al. [7] applied feature selection and RF with SMOTE on the NSL-KDD dataset for intrusion detection. Random forest is a machine learning algorithm that combines multiple decision trees to make predictions. In their method, Tesfahun et al. utilized feature reduction techniques to select the most relevant features from the dataset. SMOTE was employed as an oversampling technique to address the class imbalance in the dataset by generating synthetic samples from the minority class. The implementation was conducted on a 2.4 GHz Intel Core i5-2430M processor with 4 GB RAM, utilizing the WEKA machine learning tool version 3.6.9. The programming language used in the study are not mentioned in the information provided. To evaluate the performance of their approach, Tesfahun et al. used recall, false positive rate (FPR), and precision metrics on the NSL-KDD IDS dataset. They partitioned the 22 attacks in the NSL-KDD dataset into five classes (Normal, DoS attack, Probing attack, R2L attack, and U2R attack) and evaluated the performance metrics for each group rather than individual attacks. Recall measures the ability of the model to identify true positive instances, FPR measures the rate of falsely classified negative instances as positive, and precision quantifies the proportion of correctly classified positive instances. The authors reported achieving a precision of 0.99 and an FPR of 0, indicating a high level of accuracy in identifying positive instances and a low rate of false positives.

Lopez-Martin et al. [8] introduced the variational generative model (VGM) for intrusion detection, utilizing a variational autoencoder, a neural network capable of generating new data samples. To handle class imbalance in the NSL-KDD dataset, the authors incorporated seven variants of SMOTE and ADASYN (adaptive synthetic sampling). They evaluated the performance of the VGM using accuracy and F1 score as metrics. The VGM achieved an average accuracy of 0.7685, indicating a relatively high level of correct predictions, while the average F1 score was reported as 0.7410, demonstrating a good bal-

ance between precision and recall in the model's performance. The specific programming language used in their study was not specified.

Sun and Liu [9] proposed SMOTE_NCL, an innovative approach that combines SMOTE with neighborhood cleaning rule (NCL) for resampling the KDD CUP 99 IDS dataset. SMOTE_NCL aims to address class imbalance by generating synthetic samples using SMOTE and then applying NCL to remove noisy and boundary data points. The NCL employs the KNN algorithm to identify and remove noisy samples and borderline instances from a dataset. Although Sun and Liu demonstrated improved area under the curve (AUC) (0.8060%) on the KDD CUP 99 IDS dataset with their algorithm, they acknowledged its limitations in terms of robustness and effectiveness. The evaluation of the algorithm's performance was based on the AUC parameter, which provides an overall assessment of the classifier's ability to discriminate between positive and negative instances.

Karatas, Demir, and Sahingoz [10] aimed to enhance the performance of machine-learning-based IDS by utilizing an imbalanced and up-to-date dataset. They employed various techniques, including SMOTE, Tomek links (it provides a way to identify and address overlapping instances in a dataset, aiding in the improvement of classification performance and the handling of class imbalance), and RF classifier to address class imbalance and improve the classification accuracy. The programming language used in their study is not specified. They evaluated the performance using several metrics, including accuracy, precision, recall, and F1 score, on the imbalanced dataset. Karatas et al. reported significant improvements in the performance of the IDS, achieving an accuracy of 98.44%, precision of 98.42%, recall of 97.23%, and F1 score of 97.82%.

Yan and Han [11] introduced a local adaptive composite minority sampling algorithm (LA-SMOTE) based on the deep learning gated recurrent unit (GRU) neural network. Their approach aims to address class imbalance by utilizing KNN to select low-frequency samples. Subsequently, high-frequency samples are chosen from the selected KNN and assigned to different regions of the sample space. The GRU neural network is then employed for training the classification model. The paper focused on building a combined intrusion detection model using imbalanced learning techniques and GRU. They evaluated the performance of LA-SMOTE on the NSL-KDD dataset and considered metrics such as FPR, false alarm rate (FAR), and accuracy to assess the effectiveness of their approach. The authors reported an average detection rate of 98.904, accuracy of 99.04, and FAR of 0.134, indicating the high performance of their model in accurately detecting intrusions. However, the programming language used for implementation is not specified in the paper.

Abdallah et al. [12] applied SMOTE in conjunction with various machine learning models to develop an IDS for vehicular ad hoc networks (VANETs) based on the ToN-IoT dataset. The aim of their study was to evaluate the performance of different machine learning algorithms and determine the most effective one. They implemented their method using Python version 3.8 programming language. The authors evaluated the performance of the IDS using metrics such as accuracy, precision, recall, F1 score, and FPR. The ToN-IoT dataset was derived from a large-scale, heterogeneous IoT network specifically designed for VANETs. Abdallah et al. reported an accuracy of 0.979, recall of 0.979, precision of 0.979, and F1 score of 0.979 for their IDS model, indicating its high performance in detecting intrusions.

While SMOTE has been widely used and enhanced with different algorithms for addressing class imbalance in IDSs, these approaches often encounter challenges related to overlapping and noisy data [13]. Additionally, the discussed papers were primarily evaluated on individual IDS datasets, raising concerns about the generalizability of their methods to other datasets.

In this paper, our goal is to validate and test our proposed algorithm and framework on both the CICIDS2018 and NSL-KDD datasets, two widely recognized benchmarks in the field of IDSs, to demonstrate the generalizability and effectiveness of our novel approach.

2.2. RL

RL is a form of machine learning that comprises four key components: the agent, the action, the environment, and the reward state. In RL, the agent determines an action based on the reward obtained from the environment. This approach employs rewards and punishments as feedback to encourage desired behavior and discourage undesired behavior, respectively [14]. The model learns from these rewards and punishments through dynamic interactions with the environment.

Arturo Servin and Daniel Kudenko [15] utilized RL to detect flooding-based distributed denial of service (DDoS) attacks. They devised a simulated network environment, integrating a Q-learning algorithm to manage the injection of anomalies and establish rewards for successful anomaly detection. Nevertheless, a limitation of this research lies in the absence of guaranteed convergence to the optimal solution. The model achieved impressive performance metrics, including an accuracy of 99%, precision of 83%, recall of 100%, and specificity of 100%.

Huang et al. [16] proposed a time series anomaly detector using deep reinforcement learning (DRL) that focuses on anomaly-based detection rather than signature-based detection. Their approach aimed to detect anomalies without assuming any underlying mechanism for anomaly patterns, making it adaptable to dynamic environments. The method utilized a logical classifier, which eliminated the need for threshold tuning and simplified the implementation of the RL algorithm. The authors evaluated their approach using accuracy (100%) and recall (100%) metrics on the Yahoo benchmark datasets [17], which served as general time series datasets and were not specifically designed for IDS datasets. The Yahoo benchmark dataset is a generic and scalable anomaly detection framework called EGADS, implemented at Yahoo, which automatically monitors and alerts on large-scale time series data for various use cases. By using these datasets, Huang et al. demonstrated the effectiveness of their DRL-based anomaly detection method in a broader context. Unfortunately, the programming language used for the implementation was not explicitly mentioned in the paper.

Caminero et al. [18] proposed a framework called Adversarial Environment Reinforcement Learning (AE-RL) for implementing a classifier based on RL theory IDS datasets. The AE-RL framework consists of two agents: the classifier agent and the environment agent. RL is utilized to generate new samples from the environment agent. The main objective of AE-RL is to create an adversarial environment that actively challenges the predictions made by the classifier.

In this approach, an intelligent environment is simulated as a second agent, which generates random samples (states) from the training data and assigns rewards based on the classifier's performance with an adversarial objective. The AE-RL framework aimed to improve the classifier's ability to detect and classify anomalies in the IDS datasets. Caminero et al. implemented their method using the NSL-KDD IDS dataset and evaluated the performance using various metrics, including time, accuracy, recall, precision, and F1 score. The implementation was conducted in Python version 3.6 on a Windows 10 Pro 64-bit operating system. The authors utilized TensorFlow version 1.15.2 and Keras version 2.2.4 as the main deep learning frameworks for the implementation. It is important to note that the AE-RL framework had a limitation in that it required intelligent sample selection rather than random selection, which could potentially affect its applicability in certain scenarios.

In comparison to previous methods, the paper by Caminero et al. demonstrated superior performance with an F1 score of 0.78, accuracy of 0.7744, precision of 0.7616, and recall of 0.7744. The results demonstrated the effectiveness of the AE-RL framework in improving the detection performance of IDS systems.

Xiangyn et al. [3] proposed an adversarial RL framework with SMOTE (AESMOTE) for anomaly detection. The AESMOTE framework combined RL with class-imbalance techniques to enhance the behavior of the environment agent and achieve improved performance in anomaly detection. AESMOTE introduced a dynamic environment for

detecting anomalies in the NSL-KDD dataset by employing various oversampling and undersampling methods. In the AESMOTE framework, the environment agent can self-learn without requiring supervision. The primary objective of AESMOTE was to provide a more effective sampling algorithm for the training dataset, aiming to achieve a balanced distribution for the classifier agent. The classifier agent learns the prediction performance and selects the most challenging samples for training, thus ensuring a balanced dataset.

To evaluate the performance of AESMOTE, Xiangyn et al. conducted experiments on the NSL-KDD IDS dataset and employed metrics such as accuracy, recall, precision, F1 score, and time. Unfortunately, the authors did not explicitly mention the programming language used for implementation. Comparing AESMOTE with the AE-RL framework, Xiangyn et al. demonstrated superior performance, achieving an F1 score of 0.8243, accuracy of 0.8209, precision of 0.8411, and recall of 0.8209. These results highlighted the effectiveness of AESMOTE in enhancing the detection performance for anomaly detection in IDS datasets. In Section 5, AESMOTE is further compared with RLFOUA, which is regarded as the most recent and best-performing method for detecting attacks in IDS datasets.

3. Methodology

This section comprehensively presents the proposed RLFOUA framework and its components, along with detailed insights into the employed datasets, the novel TFRSMOTE algorithm, and the testing methodology applied to the independent sets of both CSE-CIC-IDS2018 and NSL-KDD datasets. The dialogue initiates by delving into the datasets employed in the research, elucidating their distinct attributes, followed by a comprehensive exposition that encompasses the delineation of the proposed RLFOUA framework, encompassing its constituent elements such as preprocessing, the RL algorithm and its functioning agents, the introduced TFRSMOTE algorithm, and the proposed IM. Finally, an overview of the employed testing methodology is presented.

3.1. Datasets

To evaluate the RLFOUA framework, we utilize two widely recognized datasets commonly employed in IDS research: NSL-KDD and CICIDS2018 [19]. We provide a brief overview of both datasets in this section.

3.1.1. CSE-CIC-IDS2018

The Canadian Institute for Cybersecurity Intrusion Detection System (CSE-CIC-ID2018) dataset consists of approximately 2,830,540 samples [20] of traffic that was simulated during the period between February and March 2018. Each file is approximately 200 KB [21] and contains 79 features. The data are binary-classified, with each sample labeled as either "Benign" or "Infiltration". The 28 February 2018 file, for instance, includes 540,568 samples of the Benign class and 68,462 samples of the Infiltration class (after removing outliers). The dataset encompasses various attack types, such as brute-force, Heartbleed, botnet, DoS, DDoS, web attacks, and network infiltration [21]. Several research papers have utilized this dataset to evaluate machine learning algorithms [22]. Given the large size of this dataset, training the RLFOUA framework would be time-consuming. Thus, we performed a preliminary test by selecting a random subset consisting of 250 infiltration and 7500 benign activities from the 28 February 2018 file for training and validating the RLFOUA framework within a reasonable timeframe to identify machine learning algorithms that exhibit superior performance when combined with RLFOUA. Finally, for main testing purposes, the entire dataset was used to train, validate, and test the RLFOUA framework. Consequently, all files for each reported month from the CSE-CIC-ID2018 dataset, totaling 6,493,978 samples, were divided into training, validation, and testing sets. The RLFOUA framework underwent training, validation, and testing on high-performance computing systems to generate a new model. Subsequently, its performance was assessed using an independent dataset that had been previously set aside for the purpose of evaluating the effectiveness of the RLFOUA framework.

3.1.2. NSL-KDD Dataset

The NSL-KDD dataset is a refined version of KDD-99, which was developed as part of the International Knowledge Discovery and Data Mining Tools Competition. NSL-KDD consists of 125,973 instances and includes the raw data from KDD Cup 1999, with some redundant data removed [23]. This dataset was constructed in 1999 for a competition aimed at building a predictive model capable of distinguishing between "good" and "bad" network connections. It comprises 41 features, including 38 continuous and 3 categorical variables. The dataset consists of one normal label and approximately 22 different types of attacks, including neptune, satan, ipsweep, portsweep, smurf, nmap, back, teardrop, warezclient, pod, guess_passwd, buffer_overflow, warezmaster, land, imap, rootkit, loadmodule, ftp_write, multihop, phf, perl, and spy. The normal activity class has the highest frequency with 67,343 samples, while "neptune" is the most frequent attack with 41,214 samples, and "spy" represents the least frequent attack with only two instances.

Please note that the information provided here is a general description of the attack types. For a more comprehensive understanding, it is recommended to refer to the specific literature or research papers that discuss these attack types in detail [24]. "Neptune" is a type of denial-of-service (DoS) attack aimed at overwhelming the target system with excessive traffic to render it unresponsive. "Satan" is a probing attack where the target system is actively scanned to identify vulnerabilities. The "ipsweep" involves scanning a range of IP addresses to locate live hosts on a network, and "portsweep" is similar but scans multiple machines to find vulnerable services. "Smurf" involves flooding the network with Internet Control Message Protocol (ICMP) echo to request packets, leading to a slowdown or denial of service, and "nmap" is a widely used network scanning tool. In the context of the dataset, this attack refers to the use of nmap for reconnaissance purposes. "Back" involves gaining unauthorized access to a system, typically by exploiting vulnerabilities in the network or application. "Teardrop" exploits a vulnerability in the reassembly of fragmented IP packets, causing the target system to crash or become unstable. "Warezclient" is associated with clients attempting to download or share copyrighted software or data without proper authorization. "Pod" (ping of death) attack involves sending oversized or malformed packets to the target system, causing it to crash or become unresponsive. "Quess_passwd" represents attempts to guess user passwords by systematically trying different combinations until a successful login is achieved. "Buffer_overflow" exploits a vulnerability in a program's buffer handling, allowing an attacker to overwrite memory and execute malicious code. "Warezmaster" is similar to warezclient, refers to the presence of a server that hosts unauthorized or pirated software. "Land" attack involves sending a spoofed TCP SYN packet with the source IP address and port set to the same as the target system. This causes the system to respond to itself and potentially crash. "Imap" attack targets the Internet Message Access Protocol (IMAP) service, which is used for retrieving email from a remote server. "Rootkit" is a collection of tools and techniques used by an attacker to gain unauthorized access and maintain control over a compromised system. "Loadmodule" involves loading a malicious code module into the target system's memory, enabling the attacker to execute arbitrary commands. "Ftp_write" refers to unauthorized write access to an FTP server, allowing an attacker to upload or modify files. "Multihop" involves an attacker using multiple compromised systems to carry out an attack, making it difficult to trace the origin. "Phf" targets the Common Gateway Interface (CGI) program called PHF, which is used for querying remote servers. "Perl" is associated with the execution of malicious Perl scripts on the target system. "Spy" is a spy attack represents unauthorized monitoring or surveillance activities aimed at gathers sensitive information from the target system.

3.2. RLFOUA Framework Description

The RLFOUA framework is introduced in this section, providing a detailed explanation of its components and operation. A flowchart illustrating the RLFOUA framework is presented in Figure 1. Designed as an RL environment, the RLFOUA framework involves

the collaboration of two agents working in tandem. The first agent assumes the role of selecting suitable classifiers and ascertaining the optimal training cessation point through validation dataset analysis, while the second agent, the innovative TRFSMOTE algorithm, undertakes the task of dataset resampling. The RLFOUA framework starts by subjecting the initial dataset to preprocessing, followed by the establishment of an algorithm repository, culminating in the utilization of the functional agents for RL. The following sections delve into the intricacies of each facet within the RLFOUA framework, expounding on their individual components and operation to provide an encompassing comprehension of the method's functioning.

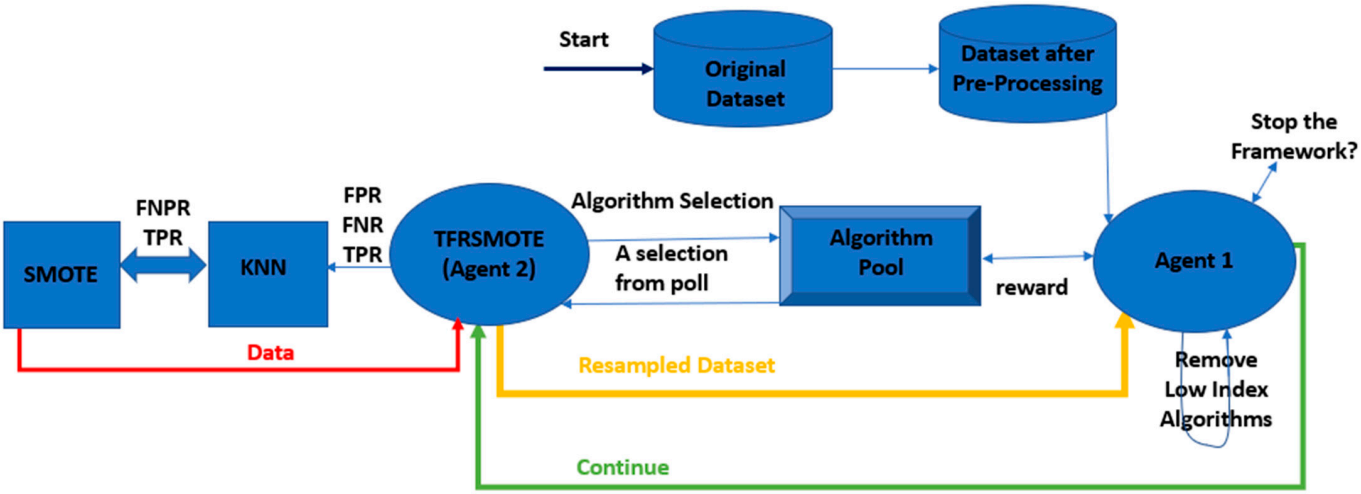

Figure 1. The proposed RLFOUA framework.

The overview of RLFOUA framework is given in Algorithm 1:

Algorithm 1: RLFOUA framework

Input: Original Dataset, A set of machine learning algorithms, oversampling algorithms, and undersampling algorithms.

Output: A new generated model for detecting attacks in an independent set of data.

1. preprocessing phase
2. Employing Functional Agents in RL
   - **a.** Establishing the Algorithm Repository
   - **b.** Enhanced RL with Functional Agents for RLFOUA
     - **i.** Call Agent 1 and Agent 2 (TFRSMOTE)

3.2.1. Preprocessing

The RLFOUA framework begins by receiving the original dataset as input. A preprocessing phase is conducted to ensure data quality and feature relevance. Initially, null data and outliers are removed, while string data are transformed into categorical numeric values. Subsequently, the forward selection algorithm [2] is employed to identify and eliminate unnecessary features. The forward selection algorithm utilizes regression analysis [2] to select the most significant features related to the label column. Variables are sequentially added to the set based on their *p*-value and R^2 value, retaining only those with a *p*-value exceeding the chosen significance level of 0.95% [25].

3.2.2. Establishing the Algorithm Repository

Following the data preprocessing stage, the dataset is ready for evaluation within the RLFOUA framework. This framework makes use of three distinct algorithm pools: machine learning algorithms, oversampling algorithms, and undersampling algorithms. The output of the algorithm will be a pipeline of a machine learning algorithm, an oversampling algorithm, and an undersampling algorithm. The pipeline serves the purpose

of combining different steps that can be tested with different settings, facilitating their sequential application to a dataset. In this context, encompassing the sequence of oversampling followed by undersampling to ultimately produce a transformed dataset for machine learning algorithms. Elaboration can be found within Algorithm 2.

Algorithm 2: Establishing the Algorithm Repository

Input: A set of machine learning algorithms, oversampling algorithms, and undersampling algorithms.

Output: A composite pipeline consisting of a singular machine learning algorithm, an oversampling algorithm, and an undersampling algorithm.

1. Set up machine learning algorithms with default parameters, including Decision Tree, Logistic Regression, Linear Discriminant Analysis, Linear SVC, Bagging Classifier, Random Forest Classifier, Extra Trees Classifier, K Neighbors Classifier, Gaussian Process Classifier, Dummy Classifier, XGB Classifier, Easy Ensemble Classifier, and Elliptic Envelope (dimension = 12).

2. Set the oversampling algorithms to include Random Oversampling, ADASYN, and SVMSMOTE (dimension = 3).

3. Set the undersampling algorithms to include Random Undersampling, Nearest Neighbor, One Sided Selection, Neighborhood Cleaning Rule, and NearMiss (dimension = 5).

4. set the actions to the following items: select one combination from the pipelines of algorithms, keep the current pipelines of algorithms, remove the pipelines of algorithms, continue the framework, stop the framework (dimension = 5).

5. Initialize Q-values for state-action pairs: Q(s, a) for all possible states (dimension = 180) and actions (dimension = 5).

6. Return a set of pipelines of algorithms from steps 1, 2, and 3 (selecting one algorithm from each step to generate a single pipeline).

3.2.3. Employing Functional Agents in RL

In this research, multiagent reinforcement learning (MARL) is used as it offers advantages over single-agent reinforcement learning (SARL) in complex scenarios [15]. MARL allows multiple agents to learn and make decisions concurrently, capturing nuanced interactions and dependencies among agents. Therefore, in the integrated RL environment of RLFOUA, two agents collaborate to optimize the algorithm selection and resampling processes. The functional agents in RL start with a preprocessed dataset. Then, they integrate it within a pipeline combining a machine learning algorithm, an oversampling method, and an undersampling technique. The goal is to produce a model capable of detecting attacks in a general dataset. In each iteration, the first agent chooses a pipeline generated from Algorithm 1, trains the machine learning algorithms after resampling within the pipeline, and calculates the reward based on the resulting performance. Concurrently, the second agent, called TFRSMOTE (proposed resampling algorithm), performs dataset resampling.

In the RLFOUA framework, it is essential to define key components of RL to better understand its operation within the context of our proposed method. These components include the following:

State: In RLFOUA, the "state" represents the current status or configuration of the learning environment, which includes the dataset and the algorithmic choices. It encapsulates all the information needed to make decisions regarding algorithm selection and resampling strategies.

Action: Actions refer to the choices available to the agents in the RLFOUA framework. Specifically, Agent 1 selects pipeline combinations, machine learning algorithms, and resampling techniques, while Agent 2 utilizes the TFRSMOTE resampling algorithm with varying parameters.

Reward: The "reward" in RLFOUA serves as a feedback signal that quantifies the quality of decisions made by Agent 1. It is computed based on the performance metrics. Rewards guide the decision-making process, enabling the selection and refinement of algorithm combinations.

Transition function: It is considered as the implicit process through which the pipeline combinations evolve. The transitions involve training and evaluating different combinations of machine learning algorithms and resampling techniques to arrive at improved selections.

Understanding these RL components within RLFOUA is pivotal for comprehending how the collaborative interaction of two agents, the Q-learning process of Agent 1, and the TFRSMOTE algorithm of Agent 2 jointly lead to optimized algorithm selection, resampling, and dataset balancing, ultimately enhancing intrusion detection system performance.

In the first phase, "Agent1", each pipeline received from the pool of machine learning, oversampling, and undersampling algorithms is trained and evaluated on the received preprocessed dataset. Evaluation metrics are computed, and IM is calculated to determine the model's performance, as described in Equation (8). Employing this metric as a pivotal guide, Agent 1 is entrusted with the pivotal task of determining whether to proceed with the framework's execution or halt it altogether. Furthermore, this metric empowers Agent 1 to make informed decisions regarding the adjustment of rewards, encompassing the crucial choices of whether to augment or diminish them. IM introduces a new strategy for comparing classification algorithms within the RLFOUA framework. The IM metric incorporates the following parameters: TP, TN, FN, and FP [2]. TP represents cases where the label is an attack, and it is correctly predicted as an attack. TN denotes instances where the label is benign, and it is correctly predicted as benign. FP refers to cases where the label is benign, but it is incorrectly predicted as an attack. FN represents situations where the label is an attack, but it is incorrectly predicted as benign. Equations (1)–(7) describe the other metrics used in this paper specifically for IM, including recall, precision, F1 score, FPR, FNR, and sensitivity.

$$\text{recall(sensitivity)} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{1}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

$$\text{F1 score} = 2 \times \frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \tag{3}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{4}$$

$$\text{FNR} = \frac{\text{FN}}{\text{TP} + \text{FN}} \tag{5}$$

$$\text{specificity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \tag{7}$$

$$\text{IM} = \frac{\sqrt{(\text{F1score}^2 \times \text{accuracy}^2 \times \text{sensitivity}^2 \times \text{specificity}^2)}}{\sqrt{\text{FPR}^2 \times \text{FNR}^2}} \tag{8}$$

The theory behind this equation is based on the practical meaning of each variable. A higher F1 score, accuracy, sensitivity, and specificity indicate better performance of the classification algorithm. Conversely, a lower FPR and FNR indicate better performance of the proposed algorithm. Since all these parameters are crucial in the RLFOUA framework, we combine them into a single equation, with F1 score, accuracy, sensitivity, and specificity in the numerator and FPR and FNR in the denominator. The IM is utilized by the first

agent in the proposed RLFOUA algorithm to decide whether to retain or discard an algorithm combination.

Following the computation of the IM by Agent 1, a comparison is made between the IM obtained from the current pipeline and the best IM derived from any of the previous iterations. In the event that enhancements are observed in the IM stemming from the current pipeline compared to the highest IM achieved in prior iterations, the model, algorithm details, metrics, and classification report are preserved. Additionally, if the IM value for the present pipeline registers an increase in comparison to the same pipeline's IM value from the previous iteration, corresponding rewards are escalated. Conversely, in cases where the IM value fails to exhibit an increase, rewards are diminished. Following this reward adjustment step, algorithm combinations possessing a reward falling below the predefined threshold (defaulted to zero in this algorithm) are excluded from the pool of algorithms available for further processing by Agent 1. This orchestrated sequence ensures the continuous refinement of the algorithm pool and model selection, based on the observed improvements in IM and rewards, in order to achieve heightened algorithmic efficacy. If we take out an algorithm combination from the pool, we keep track of its performance measures, like IM, along with its model. We perform this so we can compare it with the next combinations of algorithms. If the removed combination performs better, it is considered the top choice, and its model is saved. After running RLFOUA, we choose the model that performed the best between all iterations for testing.

In the RLFOUA framework, Agent 1 and Agent 2 employ Q-learning, a value-based reinforcement learning algorithm, to guide its decision-making process. Q-learning involves learning a value function (Q-function) that calculates the return for each state–action pair in Equation (9). At each time step, Agent 1 updates its Q-values from Equation (10) based on the immediate rewards obtained which will be used for Agent 2 as well for resampling. In these equations, the variable V_s represents the expected cumulative reward at state s, R_s represents the immediate reward (IM metric in Equation (8)) upon transitioning to state s, $P_{(s'|s,a)}$ is the probability of transitioning to state s′ from state s when taking action a, γ denotes the discount factor, which falls within the range of [0, 1], $Q_{(s,a)}$ represents the estimated value (Q-value) of taking action a in state s, and s′ is the next state. In the Bellman equation [18], when $\gamma = 0$, the reward only considers immediate outcomes R_s without accounting for future rewards. Conversely, when $\gamma = 1$, all future rewards are considered. In the RLFOUA framework, we set $\gamma = 0$ to prioritize immediate rewards of R_s, as we do not have the means to calculate future rewards.

$$V_s = R_s + \gamma \times \sum_{s'}^{n} P_{(s'|s,a)} \times V_{s'} \tag{9}$$

$$Q_{(s,a)} = Q_{(s,a)} + \left[R_s + \gamma \times \max Q_{(s',a)} - Q_{(s,a)} \right] \tag{10}$$

The second phase, "Agent2", involves the TFRSMOTE algorithm. It resamples datasets through K-nearest neighbors (KNN) selection and manipulation, aiming to refine the data for improved learning. Following the prediction of the validation data by the generated model, three distinct datasets are produced: False Negative Positive Rate (FNPR), True Positive Rate (TPR), and True Negative Rate (TNR). FNPR is a combination of FPR and FNR, containing inaccurately classified data. TPR comprises correctly classified benign data, while the remaining data fall under TNR, encompassing accurately classified infiltration data.

To apply the KNN algorithm, the value of k is set between 20 and 2000. It is dynamically reduced if the size of the selected data is less than or equal to twice k to ensure at least two random data points are selected for resampling. In the realm of oversampling undertaken by TFRSMOTE, the FNPR dataset undergoes the KNN algorithm to become clustered. The average of two samples within each group is calculated, producing new samples that are added to the FNPR dataset. The undersampling process in TFRSMOTE begins by applying the KNN algorithm to the TPR dataset, selecting k groups of samples,

and removing them. The resulting new dataset is then combined with the FNPR and TNR datasets and used for the subsequent execution of the algorithm combination training by Agent 1 in RLFOUA. This step effectively removes unimportant nonattack data rows that were correctly identified and oversamples important data that were falsely detected, resulting in a balanced dataset. The TFRSMOTE process is completed at this stage, but it is repeated for each step within an iteration of the RLFOUA framework.

In summary, the RL algorithm in RLFOUA framework, in conjunction with the TFRSMOTE algorithm and IM, facilitates the collaborative interaction between two agents. This interaction leads to the selection of optimal machine learning and resampling algorithms for dataset training, while simultaneously ensuring effective oversampling and undersampling to enhance performance. The process of TFRSMOTE is repeated for each step in an iteration of the RLFOUA framework, thereby removing unimportant nonattack data rows while oversampling relevant falsely detected data to achieve dataset balance.

The enhanced RL with functional agents for RLFOUA algorithm is illustrated in Algorithm 3, offering a comprehensive and detailed overview of its procedural steps.

Algorithm 3: Enhanced RL with functional agents for RLFOUA

Input: A preprocessed dataset with a composite pipeline consisting of a singular machine learning algorithm, an oversampling algorithm, and an undersampling algorithm.
Output: A new generated model for detecting attacks in an independent set of data
Initialize Q-table with zeros.
Set reward discount factor (γ) to 0 for Bellman Equation (prioritizing immediate rewards).
State: Composite pipeline (ML algorithm, oversampling, undersampling)
Action: Selecting an algorithm combination
Reward: IM metric, based on F1 score, accuracy, FPR, FNR, specificity, sensitivity
Transition: Updating Q-values **do**
1. For each pipeline received from Algorithm 2 (Agent1):
1.1. Utilize the given pipeline to train on the training dataset and subsequently make predictions on the validation dataset.
1.2. Calculate evaluation metrics such as F1 score, accuracy, FPR, FNR, specificity, sensitivity, and time.
1.3. Calculate IM $= \frac{\sqrt{(\text{F1score}^2 \times \text{accuracy}^2 \times \text{sensitivity}^2 \times \text{specificity}^2)}}{\sqrt{\text{FPR}^2 \times \text{FNR}^2}}$.
1.4. If the IM is better than the best IM from any of the previous iterations:
1.4.1. Save the model, algorithm details, metrics, and classification report.
1.5. If the IM for this pipeline is improved compared with the same pipeline in previous iteration:
1.5.1. Increase the reward for this pipeline.
1.6. Else:
1.6.1. Decrease the reward for this pipeline.
1.7. Update the Q-values in reward table using the Bellman equation.
1.8. TFRSMOTE algorithm (Agent2): Set the value of k for KNN between 20 and 2000. It is dynamically reduced if the size of the selected data is less than or equal to twice k to ensure at least two random data points are selected for step 1.9.5.
1.9. For value of k:
1.9.1. Predict the validation data from the generated model from step 1.4.1
1.9.2. Calculate the FPR, FNR, and TPR data from predicting the generated model from RLFOUA on the validation set.
1.9.3. Create three datasets: FNPR, which merges FPR and FNR, containing falsely classified data, TPR, containing correctly classified benign data, and remaining data.
1.9.4. Apply the KNN algorithm to select k groups of samples from the FNPR dataset obtained in step 1.9.3.
1.9.5. Calculate the average of two randomly selected samples from each group of the KNN results and add them to the FNPR dataset as new samples.
1.9.6. Apply KNN to select k groups of samples from the TPR dataset obtained in step 1.9.3.
1.9.7. Select one element randomly from each group.
1.9.8. Remove the elements selected in step 1.9.7 from the TPR dataset.
1.9.9. Generate a new dataset by combining the new TPR, new FNPR, and the remaining data from the original dataset obtained in step 1.9.3
1.10. replace the dataset with the new resampled dataset from the TFRSMOTE algorithm for the next iteration.
2. Exclude the algorithm combinations with rewards lower than the threshold from further processing. By default, this threshold is set to zero in this algorithm. However, retain its model and performance measures for later comparison with other algorithm combinations (Agent1).
While IM shows improvement in at least one of the algorithm combinations, compared to previous loop iteration.
Return the new balanced dataset and the generated model.

3.2.4. Testing Method

To establish the efficacy and generalizability of the proposed algorithm, comprehensive tests are conducted using two distinct types of IDS datasets: CSE-CIC-ID2018 (binary-class) and NSL-KDD (multiclass). These datasets are representative of different scenarios, ensuring that the algorithm's performance extends beyond specific dataset characteristics. The evaluation results differ between the two datasets due to their distinct characteristics.

The testing process follows a systematic procedure, outlined as follows:

1. Preliminary application of RLFOUA framework on CSE-CIC-IDS2018 dataset: The smaller section of the dataset, as discussed in Section 3.1.1, undergoes division into training and validation subsets. Within the RLFOUA framework, machine learning algorithms are trained using the training set, while the validation dataset is utilized to compute classification metrics. This framework's impact is assessed by comparing pre- and post-application metrics, encompassing F1 score, accuracy, recall, and precision. The four most proficient algorithm combinations are ranked independently for each dataset, effectively highlighting the RLFOUA framework's capacity to enhance classification performance.
2. Preliminary application of RLFOUA framework on NSL-KDD dataset: Similar analyses are conducted on the entire NSL-KDD datasets, following the same partitioning approach and evaluating performance based on distinct training and validation sets. Subsequent ranking of the top four algorithm combinations underscores the consistent advancement in classification performance through the RLFOUA framework.
3. Primary testing on independent set of CSE-CIC-IDS2018 dataset: In order to corroborate the generalizability of the RLFOUA framework, independent datasets sourced from the CSE-CIC-ID2018 dataset are engaged. The entire dataset undergoes tripartite division into training, validation, and independent testing subsets. The framework is trained and validated on these data subsets, yielding a new model for subsequent testing. This newly formed model is then evaluated on independent testing sets using the RLFOUA framework, affirming the framework's reliability and ability to maintain consistent performance beyond training and validation phases.
4. Primary testing on independent set of NSL-KDD dataset: To extend the applicability of the method, the approach employed in test 3 is replicated on the NSL-KDD datasets. Employing a tripartite division, the datasets are split into training, validation, and independent testing subsets, with the framework trained and validated accordingly. The resultant model is then subjected to evaluation on separate testing sets, reinforcing the method's generalizability and its capacity to ensure stable performance beyond training and validation stages.

Through this comprehensive testing process, the effectiveness and robustness of the RLFOUA framework, along with the TFRSMOTE resampling algorithm, will be determined. In addition, their superiority over other existing methods will be ascertained. Enhanced performance will be evident in the form of higher F1 scores, accuracy, recall, and precision, establishing the algorithm's efficacy for real-world IDS applications.

4. Results

In this section, we evaluate the performance of the framework using two distinct datasets, namely, CSE-CIC-IDS2018 and NSL-KDD, and demonstrate the improvements achieved in terms of classification metrics. Furthermore, we test the RLFOUA framework on an independent dataset to assess its generalizability. The application of the RLFOUA framework with TFRSMOTE algorithm on the CSE-CIC-IDS2018 and NSL-KDD datasets reveals notable enhancements in the classification metrics. The evaluation results differ between the two datasets due to their distinct characteristics.

4.1. Preliminary Application of RLFOUA Framework on CSE-CIC-IDS2018 Dataset

The RLFOUA framework, which compares different sequences and combinations of machine learning algorithms and resampling algorithms, identified the top-performing

algorithm combinations, listed in Table 1 in descending order. The results shown in this table highlight the precision, recall, and F1 score for each algorithm combination, categorized by the data sample classification (benign and infiltration). The RLFOUA framework determines that the algorithm combination of Balanced Bagging Classifier, Random Oversampling, Random Undersampling, and TFRSMOTE achieves the highest performance with an F1 score of 0.9998 for the "benign" class and 0.9784 for the "infiltration" class. This is followed by the Random Forest Classifier, which achieves an F1 score of 0.9999 for the "benign" class and 0.9910 for the "infiltration" class. The Bagging Classifier and Extra Trees Classifier also exhibit strong performance, although the Bagging Classifier has relatively lower precision for the "infiltration" class.

Table 1. Classification report for the validation set of CSE-CIC-IDS2018 dataset, comprising different algorithm and sampling technique combinations within the RLFOUA framework for infiltration attack, arranged in descending order of performance metrics.

Algorithm	Attack Type	Precision	Recall	F1 Score	Support	Accuracy
Balanced Bagging Classifier Random Oversampling Random Undersampling TFRSMOTE	Benign	1	0.9996	0.9998	7322	0.9996
	Infiltration	0.9577	1	0.9784	68	
Random Forest Classifier Random Oversampling Neighborhood Cleaning Rule TFRSMOTE	Benign	1	0.9999	0.9999	7308	0.9999
	Infiltration	0.9821	1	0.991	55	
Extra Trees Classifier ADASYN Neighborhood Cleaning Rule TFRSMOTE	Benign	0.9999	0.9986	0.9992	7308	0.9985
	Infiltration	0.8485	0.9836	0.9106	57	
Bagging Classifier ADASYN Neighborhood Cleaning Rule TFRSMOTE	Benign	0.9999	0.9966	0.9982	7322	0.9965
	Infiltration	0.7059	0.9825	0.8219	61	

The exceptional performance of certain algorithm combinations, notably the Balanced Bagging Classifier with specific resampling techniques, can be attributed to their ability to effectively handle the complex nature of the CSE-CIC-IDS2018 dataset. This dataset exhibits a considerable class imbalance between benign and infiltration samples, making it challenging for traditional algorithms to discern between the two classes accurately. The combination of techniques employed, including random oversampling, random undersampling, and TFRSMOTE, work synergistically to address this imbalance. Random oversampling augments the minority class, ensuring that it contributes meaningfully to the learning process. Conversely, random undersampling helps mitigate the dominance of the majority class, preventing it from overshadowing the learning process. Additionally, TFRSMOTE fine-tunes the sampling process, adapting dynamically to the dataset's characteristics. This strategic combination of techniques results in a more balanced and representative training set, enabling the algorithms to make informed decisions and achieve exceptional performance in detecting both benign and infiltration attacks. The effectiveness of these combinations underscores the significance of tailored resampling strategies in enhancing intrusion detection systems, particularly in scenarios with imbalanced datasets.

4.2. Preliminary Application of RLFOUA Framework on NSL-KDD Dataset

In order to assess the generalizability of the RLFOUA framework across different datasets, we apply it to the NSL-KDD dataset and present a summary of the results in Ta-

ble 2. The table displays the classification report of the RLFOUA framework, incorporating four of the best-performing algorithms along with TRFSMOTE. The algorithm combinations are ranked in order of their performance, with the top-performing combinations listed in the table.

Table 2. Classification report for the validation set of NSL-KDD dataset, comprising different algorithm and sampling technique combinations within the RLFOUA framework for infiltration attack, arranged in ascending order of performance metrics.

Algorithm	Attack Type	Precision	Recall	F1 Score	Support	Frequency	Accuracy
Decision Tree Classifier Random Oversampling Algorithm Near Miss Undersampling TFRSMOTE	Neptune	0.9999	0.9991	0.9995	16,585	41,214	
	Nmap	0.9685	0.9898	0.979	590	1493	
	Normal	0.9982	0.9987	0.9985	26,709	67,343	
	Rootkit	0.0694	1	0.1299	5	10	
	Multihop	1	1	1	7	7	
	Weighted Avg	0.9885	0.9847	0.9847	50,378		0.9847
Bagging Classifier Random Oversampling Algorithm Near Miss Undersampling TFRSMOTE	Neptune	0.9999	0.9993	0.9996	16,579	41,214	
	Nmap	0.9662	0.9967	0.9812	603	1493	
	Normal	0.9994	0.9994	0.9994	26,828	67,343	
	Rootkit	0.75	1	0.8571	3	10	
	Multihop	1	1	1	4	7	
	Weighted Avg	0.9982	0.9981	0.9981	50,390		0.9981
XGB Classifier Random Oversampling Algorithm Neighborhood Cleaning Rule Undersampling TFRSMOTE	Neptune	0.9999	0.9999	0.9999	16,366	41,214	
	Nmap	0.9982	0.993	0.9956	568	1493	
	Normal	1	0.9993	0.9996	27,123	67,343	
	Rootkit	0.5714	1	0.7273	4	10	
	Multihop	1	1	1	2	7	
	Weighted Avg	0.9996	0.9995	0.9995	50378		0.9995
Extra Trees Classifier Random Oversampling Algorithm Neighborhood Cleaning Rule Undersampling TFRSMOTE	Neptune	0.9999	1	1	16,462	41,214	
	Nmap	1	0.9951	0.9975	613	1493	
	Normal	0.9999	0.9994	0.9996	26,852	67,343	
	Rootkit	0.8	0.8	0.8	5	10	
	Multihop	1	1	1	4	7	
	Weighted Avg	0.9996	0.9995	0.9995	50,420		0.9995

Table 2 presents the classification results of the RLFOUA framework using the four algorithms along with their respective sampling techniques. The metrics included in the table are precision, recall, F1 score, support, frequency, and accuracy. In conclusion, the application of the RLFOUA framework on the NSL-KDD dataset demonstrates its generalizability to different types of datasets. The results demonstrate that the RLFOUA framework, by autonomously identifying the top-performing algorithm combinations, effectively detects various attack types with high precision, recall, and F1 scores. Among the evaluated algorithms, the Extra Trees Classifier stands out as the most promising choice for detecting attacks within the NSL-KDD dataset. These findings contribute valuable insights for selecting suitable algorithms and sampling techniques when applying the RLFOUA framework in real-world intrusion detection scenarios.

The remarkable effectiveness of certain algorithm combinations on the NSL-KDD dataset can be attributed to their adaptability to the dataset's diverse range of attack types. The Extra Trees Classifier, for instance, demonstrated exceptional performance, likely due to its inherent ability to handle complex decision boundaries and high-dimensional feature spaces. Additionally, the integration of Random Oversampling and Neighborhood Cleaning Rule Undersampling further enhanced the model's ability to discern subtle patterns within the data, contributing to higher precision and recall scores. This combination of algorithms and resampling techniques effectively leveraged the dataset's unique characteristics, allowing the RLFOUA framework to autonomously identify and employ the most suitable strategies for detecting various attack types.

4.3. Primary Testing on Independent Set for CSE-CIC-IDS2018

In this section, we test the RLFOUA framework on the CSE-CIC-IDS2018 dataset. The best models obtained from the RLFOUA framework in Section 4.1 are evaluated using the testing set as an independent dataset. The testing results are presented in Table 3, which demonstrates the effectiveness of the RLFOUA framework on the CSE-CIC-IDS2018 dataset.

Table 3. Classification metrics on the testing set of CSE-CIC-IDS2018 dataset using RLFOUA framework.

	Precision	Recall	F1 Score	Support
Benign	0.9953	0.986	0.9907	538,865
Infiltration	0.9348	0.9774	0.9556	110,453
Accuracy			0.9846	649,318
Macro Avg	0.9651	0.9817	0.9731	649,318
Weighted Avg	0.985	0.9846	0.9847	649,318

Table 3 provides a comprehensive overview of the classification metrics achieved after applying the RLFOUA framework on the independent testing set of the CSE-CIC-IDS2018 dataset. Macro Avg displays the macro-average metrics, which calculate the average of precision, recall, and F1 score, respectively, across all classes. It can provide insights into the model's performance, considering class imbalances. Weighted Avg metrics consider the class support (number of instances) in the calculation, providing a more representative performance evaluation, particularly when dealing with class imbalances. In the context of intrusion detection, precision reflects the system's ability to accurately flag potential threats without generating unnecessary alarms. Recall, on the other hand, assesses the proportion of true positives correctly identified from all actual positives in the dataset. It is particularly important in security applications to minimize false negatives, ensuring that actual threats are not overlooked. The F1 score strikes a balance between precision and recall, offering a combined metric that considers both false positives and false negatives. A higher F1 score indicates a model that excels in both precision and recall, signifying a robust detection capability. The results indicate an accuracy of 0.9846, a macro average recall of 0.9817, a weighted average recall of 0.9846, a macro average F1 score of 0.9731, a weighted average F1 score of 0.9847, a macro average precision of 0.9651, and a weighted average precision of 0.985.

4.4. Primary Testing on Independent Set for NSL-KDD

Within this section, an assessment of the RLFOUA framework is conducted using the NSL-KDD dataset. The superior models garnered from the framework, as elucidated in Section 4.2, undergo evaluation on an autonomous testing set. The outcomes of this evaluation, encapsulated in Table 4, substantiate the efficacy of the RLFOUA framework when applied to the NSL-KDD dataset. Table 4 presents the classification metrics for the testing of the RLFOUA framework on the NSL-KDD dataset, further testing its performance on a different dataset. The RLFOUA framework undergoes evaluation across various attack types, providing corresponding precision, recall, and F1 score values.

Table 4. Classification metrics for testing the RLFOUA framework with TFRSMOTE algorithm on NSL-KDD dataset. Support parameter indicates the frequency of each class in the independent set of the dataset, while frequency represents the occurrence of each class in the entire dataset.

Attack Type	Precision	Recall	F1 Score	Support	Frequency
Back	1	0.9873	0.9936	79	956
Buffer_overflow	1	0.8	0.8889	5	30
Ftp_write	0	0	0	1	8
Guess_passwd	1	1	1	8	53
Ipsweep	0.9945	0.9973	0.9959	365	3599
Land	0.5	1	0.6667	1	18

Table 4. *Cont.*

Attack Type	Precision	Recall	F1 Score	Support	Frequency
Loadmodule	0	0	0	1	9
Multihop	1	1	1	1	7
Neptune	1	1	1	4147	41,214
Nmap	0.986	0.986	0.986	143	1493
Normal	0.9976	0.9996	0.9986	6713	67,343
Pod	1	1	1	18	201
Portsweep	0.9963	1	0.9982	271	2931
Rootkit	0	0	0	2	10
Satan	1	0.9864	0.9931	367	3633
Smurf	1	1	1	283	2646
Teardrop	1	1	1	95	892
Warezclient	0.978	0.9271	0.9519	96	890
Warezmaster	1	1	1	1	20
Accuracy			0.9981	12,597	
Macro Avg	0.8133	0.8255	0.8144	12,597	
Weighted Avg	0.9978	0.9981	0.9979	12,597	

The RLFOUA framework demonstrates promising results on the NSL-KDD dataset, effectively identifying and classifying different types of attacks. Comparing the metrics across different attack types, we can observe variations in the model's performance. The aggregated metrics at the bottom of the table provide an overview of the overall performance. The accuracy of the model is 0.9981, indicating a high overall accuracy in classifying the attack types. The macro average and weighted average metrics calculate the average precision, recall, and F1 score across all attack types. The macro average gives equal weight to each attack type, while the weighted average considers the support (number of instances) of each attack type when calculating the average.

Table 4 offers valuable insights into the performance of the IDS model for different attack types, enabling a comprehensive assessment of its effectiveness in detecting and classifying network intrusions. The overall accuracy achieved by the RLFOUA framework on the NSL-KDD dataset was 0.9981, demonstrating its effectiveness in accurately classifying attack types. The macro average metrics, with precision, recall, and F1 score values of approximately 0.81, reflect an acceptable overall performance across all attack types, considering their varying support and frequency in the dataset.

Upon examining the results for different attack types in Table 4, several noteworthy trends and variations emerge. The framework demonstrates exceptional proficiency in detecting certain types of attacks, such as "neptune", "guess_passwd", "multihop", and "pod", achieving perfect precision, recall, and F1 score values. These results suggest that the RLFOUA framework, in tandem with the TFRSMOTE algorithm, excels in accurately identifying these specific attack categories. However, it is important to note that for less common attack types, like "ftp_write", "loadmodule", "rootkit", and "warezmaster", the performance metrics indicate room for improvement. The precision and recall values for these categories are relatively lower, indicating a potential area for further refinement in the framework's ability to discern these less-frequent attack types. These observations highlight the framework's strengths in handling prevalent attack categories, while also identifying potential areas for enhancement, particularly in the context of rare or infrequent attacks. This balanced evaluation underscores the framework's potential to contribute significantly to the field of intrusion detection and bolster cybersecurity measures.

5. Discussion

In this paper, we introduce RLFOUA, an RL framework, which is seamlessly integrated with a novel resampling algorithm known as TFRSMOTE. Our comprehensive approach showcases remarkable performance across two well-known IDS datasets: CSE-CIC-ID2018

and NSL-KDD. The former dataset, CSE-CIC-ID2018, is structured as a binary classification task involving benign and infiltration instances. On the other hand, NSL-KDD encompasses a diverse spectrum of attack types, including those occurring at a frequency of fewer than ten instances within the dataset. While these datasets are widely acknowledged in the IDSs field, they offer distinct features and encompass a variety of attack characteristics. It is noteworthy that, unlike many other papers, which concentrate solely on one dataset, our proposed methodology is rigorously evaluated on both CSE-CIC-ID2018 and NSL-KDD datasets, thus showcasing its robustness and generalizability across diverse data scenarios. This comprehensive evaluation substantiates the superior accuracy, precision, recall, and F1 score achieved by our method.

As depicted in Table 2 within Section 4, for the "neptune" attack type, all four algorithms achieve high precision, recall, and F1 scores, indicating their effectiveness in detecting this attack type. The Extra Trees Classifier achieves perfect recall and F1 score, indicating that it captures all instances of "neptune" attacks without any false negatives. The "nmap" attack type is also well detected by all algorithms, with the Bagging Classifier and XGB Classifier demonstrating slightly higher recall and F1 scores compared to the Decision Tree Classifier and Extra Trees Classifier. All algorithms accurately classify instances labeled as "normal" with high precision, recall, and F1 scores, indicating their ability to generalize well for benign network activity. The Decision Tree Classifier exhibits the lowest performance for the "rootkit" and "multihop" attack types, particularly in terms of recall and F1 score. On the other hand, the Bagging Classifier, XGB Classifier, and Extra Trees Classifier show better recall and F1 scores for these categories. Overall, the Extra Trees Classifier achieves the highest recall and F1 scores for most attack types, making it particularly effective in detecting a wide range of attacks within the NSL-KDD dataset.

The weighted average scores provide an overview of the algorithms' overall performance across all attack types, considering their frequency in the dataset. The Bagging Classifier and XGB Classifier exhibit higher weighted average F1 scores, indicating their balanced performance across the entire dataset. Moreover, referring to Table 4 within Section 4, the application of RLFOUA on the independent NSL-KDD dataset reveals intriguing insights. Particularly, the "nback" attack type stands out with a precision of 1.000, indicating the precise classification of all instances associated with this attack type. However, the "buffer_overflow" attack type indicates that while the precision is perfect (1), only 80% of actual "buffer_overflow" instances were captured (recall of 0.8), resulting in an F1 score of 0.8889. Some attack types, such as "ftp_write" and "loadmodule", have precision, recall, and F1 score of 0, indicating that the model failed to correctly classify any instances for these attack types.

Transitioning to the next segment, a comparative analysis between RLFOUA and AESMOTE is initiated to discern their unique attributes. AESMOTE, considered one of the most recent and high-performing algorithms in the domain, is marked by a set of unique attributes. Both RLFOUA and AESMOTE operate within an RL framework, leverage the NSL-KDD dataset, and adopt accuracy, recall, precision, and F1 score as metrics for assessing performance. The selection of AESMOTE as a benchmark is predicated on its exceptional performance relative to other contemporaneous methodologies.

The main differences between RLFOUA and AESMOTE lie in the resampling approach, dataset consideration, algorithm selection, and performance metrics. RLFOUA introduces the innovative TFRSMOTE algorithm, which systematically generates synthetic data by oversampling falsely detected instances and undersampling correctly detected benign activities. In contrast, AESMOTE relies on the widely used SMOTE algorithm for resampling.

Another distinction is that RLFOUA integrates a wide range of 23 machine learning algorithms, 8 undersampling techniques, and 3 oversampling methods, in addition to TFRSMOTE. This comprehensive exploration of the solution space allows for a more comprehensive analysis. In comparison, AESMOTE primarily employs SMOTE as its re-

sampling method and does not explicitly incorporate multiple machine learning algorithms or undersampling techniques.

Furthermore, RLFOUA evaluates performance using various metrics, including accuracy, precision, recall, and F1 score, on both CSE-CIC-ID2018 and NSL-KDD datasets, while AESMOTE only focuses on the NSL-KDD dataset. Additionally, RLFOUA utilizes a new metric, IM, for comparing algorithms. In contrast, AESMOTE does not specify a new metric used for performance evaluation.

Table 5 provides a comprehensive comparative analysis of the performance metrics between RLFOUA and AESMOTE. In the case of the NSL-KDD dataset, RLFOUA's testing set comprises 12,598 independent instances, carefully extracted from the original file. In contrast, AESMOTE is divided into a training set with 125,973 samples and a testing set with 22,544 samples [3]. The table showcases key metrics, including F1 score, accuracy, precision, and recall, for both methods. As illustrated in Table 5, RLFOUA consistently outperforms AESMOTE across all metrics, underscoring its superior performance. This distinction is particularly evident in the significantly higher F1 score, accuracy, and recall values achieved by RLFOUA.

Table 5. Comparative performance metrics of RLFOUA and AESMOTE.

Method	F1 Score	Accuracy	Precision	Recall
RLFOUA	0.9061	0.9981	0.9055	0.9118
AESMOTE	0.8243	0.8209	0.8411	0.8209

In conclusion, this paper introduces RLFOUA, an innovative approach that incorporates the novel TFRSMOTE resampling algorithm. We rigorously validated and tested RLFOUA on multiple datasets, employing a diverse range of machine learning algorithms and undersampling techniques. The performance metrics presented here serve as a testament to the effectiveness of our approach. The substantial differences in performance between AESMOTE and RLFOUA underscore the unique contributions of our work and its potential to advance the field of anomaly detection in intrusion detection systems.

Beyond the realm of intrusion detection, the RLFOUA framework, coupled with the innovative TFRSMOTE resampling algorithm, holds significant promise for applications in other domains characterized by imbalanced datasets and the need for robust anomaly detection. Industries ranging from finance to healthcare and beyond stand to benefit from the heightened accuracy and precision demonstrated by RLFOUA. By effectively addressing the challenges posed by class imbalances in data, our approach opens avenues for the development of more reliable and efficient anomaly detection systems across diverse domains. Moreover, the adaptability of RLFOUA, demonstrated through its performance across multiple datasets, suggests its potential for customization to suit specific application scenarios, further expanding its applicability in real-world settings.

6. Conclusions

IDSs are essential for identifying and mitigating malicious activities within networks. The application of machine learning algorithms to imbalanced IDS datasets can significantly enhance the performance of existing IDS methods. However, the challenge lies in the imbalance of such datasets, where malicious activities can easily conceal themselves among a large volume of normal events. Consequently, machine learning algorithms may prioritize recognizing normal events, potentially missing critical attacks. Therefore, there is a pressing need for algorithms that can effectively balance IDS datasets, leading to better-trained machine learning models.

In this paper, we present RLFOUA, an RL framework specifically designed to address the imbalanced nature of IDS datasets. RLFOUA is enhanced through the integration of a novel resampling algorithm called TFRSMOTE. TFRSMOTE facilitates dataset balance

for machine learning algorithm training by systematically oversampling falsely detected instances and undersampling correctly detected benign activities.

We tested the RLFOUA framework on two widely used IDS datasets: CSE-CIC-IDS2018 and NSL-KDD. The application of RLFOUA on the independent set from the CSE-CIC-IDS2018 dataset yielded impressive results, with precision of 0.985%, recall of 0.9846%, accuracy of 0.9846%, and F1 score of 0.9847%. Similarly, when applied to the independent set from the NSL-KDD dataset, RLFOUA achieved precision of 0.9055%, recall of 0.9118%, accuracy of 0.9981%, and F1 score of 0.9061%. These results were compared with the best and latest tested algorithms, AESMOTE, which were applied to the same datasets using the same performance metrics. As outlined in Section 5, there is an average enhancement of 11% in recall, 7.6% in precision, 21.5% in accuracy, and 9.9% in F1 score when compared to AESMOTE. The comparison revealed significant performance improvements obtained by the RLFOUA framework.

Overall, the RLFOUA framework presents a promising environment for effectively balancing imbalanced IDS datasets. Through the incorporation of the TFRSMOTE algorithm into the RLFOUA framework, we demonstrated the capability to address the challenges posed by imbalanced datasets in machine-learning-based IDS applications. The results obtained underscore the potential of RLFOUA in enhancing intrusion detection capabilities and contributing to the development of robust cybersecurity measures. Looking ahead, further research avenues could explore the fine-tuning of RLFOUA's algorithms and resampling techniques to adapt to evolving threat landscapes. Additionally, investigating the integration of RLFOUA with emerging network security technologies and exploring its performance in real-time scenarios are promising directions for future work. These endeavors aim to continue advancing the field of anomaly detection in intrusion detection systems and fortifying cybersecurity measures.

Author Contributions: Conceptualization, N.A. and M.J.; Methodology, N.A.; Software, N.A.; Validation, N.A. and M.J.; Formal analysis, N.A. and M.J.; Investigation, N.A. and M.J.; Resources, N.A. and M.J.; Data curation, N.A.; Writing—original draft, N.A.; Writing—review & editing, N.A. and M.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: Suggested Data Availability Statements are available in https://www.unb.ca/cic/datasets/index.html.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Abedzadeh, N.; Jacobs, M. Using Markov Chain Monte Carlo Algorithm for Sampling Imbalance Binary IDS Datasets. In Proceedings of the 12th International Workshop on Security, Privacy, Trust for Internet of Things (IoTSPT) at the ICCCN 2022, Honolulu, HI, USA, 25–28 July 2022; pp. 1–6.
2. Abedzadeh, N.; Jacobs, M. A Survey in Techniques for Imbalanced IDS Datasets. In Proceedings of the ICICCS 2022: 16th International Conference on Intelligent Computing and Control Systems, Madurai, India, 25–27 May 2022; pp. 1–6.
3. Ma, X.; Shi, W. AESMOTE: Adversarial Reinforcement Learning with SMOTE for Anomaly Detection. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 790–802. [CrossRef]
4. Phetlasy, S.; Ohzahata, S.; Wu, C.; Kato, T. Applying SMOTE for a Sequential Classifiers Combination Method to Improve the Performance of Intrusion Detection System. In Proceedings of the IEEE International Conference on Dependable, Autonomic and Secure Computing, Fukuoka, Japan, 5–8 August 2019; pp. 1–6.
5. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Intell. Res.* **2022**, *16*, 321–357. [CrossRef]
6. Qazi, N.; Raza, K. Effect of feature selection, synthetic minority over-sampling (SMOTE), and under-sampling on class imbalance classification. In Proceedings of the 14th International Conference on Modelling and Simulation, Cambridge, UK, 28–30 March 2012; pp. 145–150.
7. Tesfahun, A.; Bhaskari, D.L. Intrusion detection using random forests classifier with SMOTE and feature reduction. In Proceedings of the International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, Pune, India, 15–16 November 2013; pp. 127–132.

8. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A. Variational data generative model for intrusion detection. *Knowl. Inf. Syst.* **2019**, *60*, 569–590. [CrossRef]
9. Sun, Y.; Liu, F. SMOTE-NCL: A re-sampling method with filter for network intrusion detection. In Proceedings of the 2nd IEEE International Conference on Computer and Communications, Sanya, China, 27–29 December 2016; pp. 1157–1161.
10. Karatas, G.; Demir, O.; Sahingoz, O.K. Increasing the Performance of Machine Learning-Based IDSs on an Imbalanced and Up-to-Date Dataset. *IEEE Access* **2020**, *8*, 32150–32162. [CrossRef]
11. Yan, B.; Han, G. LA-GRU: Building combined intrusion detection model based on imbalanced learning and gated recurrent unit neural network. *Secur. Commun. Netw.* **2018**, *2018*, 1–13. [CrossRef]
12. Gad, A.R.; Nashat, A.A.; Barkat, T.M. Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset. *IEEE Access* **2012**, *9*, 142206–142217. [CrossRef]
13. Jimoh, I.A.; Ismaila, I.; Olalere, M. Enhanced Decision Tree—J48 With SMOTE Machine Learning Algorithm for Effective Botnet Detection in Imbalanced Dataset. In Proceedings of the15th International Conference on Electronics Computer and Computation, Abuja, Nigeria, 10–12 December 2019; pp. 1–6.
14. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; The MIT Press: Cambridge, MA, USA, 2018.
15. Servin, A.; Kudenko, D. Multi-Agent Reinforcement Learning for Intrusion Detection. Ph.D. Thesis, University of York, York, UK, 2009.
16. Huang, C.; Wu, Y.; Zuo, Y.; Pei, K.; Min, G. Towards experienced anomaly detector through reinforcement learning. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 1–6.
17. Laptev, N.; Amizadeh, S.; Flint, I. Generic and scalable framework for automated time-series anomaly detection. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'15, Sydney, Australia, 10–13 August 2015; pp. 1939–1947.
18. Caminero, G.; Lopez-Martin, B.; Carro, M.; Sanchez-Esguevillas, A. Adversarial environment reinforcement learning algorithm for intrusion detection. *Comput. Netw.* **2019**, *159*, 96–109. [CrossRef]
19. Vij, C.; Saini, H. Intrusion Detection Systems: Conceptual Study and Review. In Proceedings of the 2021 6th International Conference on Signal Processing, Computing and Control, Xi'an, China, 9–11 April 2021; pp. 694–700.
20. Maseer, Z.K.; Yusof, R.; Bahaman, N.; Mostafa, S.A.; Foozy, C.F.M. Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. *IEEE* **2021**, *9*, 1–6. [CrossRef]
21. A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018). Available online: https://registry.opendata.aws/cse-cicids2018 (accessed on 1 January 2022).
22. Mbow, M.; Koide, H.; Sakurai, K. An Intrusion Detection System for Imbalanced Dataset Based on Deep Learning. In Proceedings of the 2021 9th International Symposium on Computing and Networking, Matsue, Japan, 23–26 November 2021; pp. 38–47.
23. Gopalan, S.S.; Ravikumar, D.; Linekar, D.; Raza, A.; Hasib, M.; Roy, B.K. Balancing Approaches towards ML for IDS: A Survey for the CSE-CIC IDS Dataset. In Proceedings of the 2020 International Conference on Communications, Signal Processing, and Their Applications, Sharjah, United Arab Emirates, 16–18 March 2021; pp. 1–6.
24. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Nashville, TN, USA, 30 March–2 April 2009; pp. 1–6.
25. Abedzadeh, N.; Jacobs, M. GANMCMCRO: A Generative Adversarial Network Markov Chain Monte Carlo Random Oversampling Algorithm for Imbalance Datasets. In Proceedings of the DMMLACS—3rd International Special Session on Data Mining and Machine Learning Applications for Cyber Security, Rome, Italy, 15–17 November 2023; pp. 1–8.