New Generation Computing (2026) 44:13
https://doi.org/10.1007/s00354-026-00318-8

RESEARCH PAPER

A Generative AI Method for Minority Class Handling in Anomaly Detection with Drift and Explainability Analysis

Kelvin J. Mwiga[1,2] · Mussa A. Dida[1] · Ahmad Mohsin[3] · Iqbal H. Sarker[3]

Received: 18 September 2024 / Accepted: 14 January 2026
© The Author(s) 2026

Abstract
Artificial Intelligence, particularly machine learning (ML) algorithms, plays a crucial role in detecting cyberattacks, including anomalies and intrusions. However, machine learning models trained on imbalanced cybersecurity datasets often struggle to accurately detect minority data instances and potential threats, thereby weakening overall system security. Despite extensive research, a persistent challenge is the inadequate explanation for model predictions concerning minority data classes. This study aims to address these limitations by developing a generative AI-based approach to manage minority classes in anomaly detection, incorporating concept drift handling and explainability analysis. We introduce an over-sampling technique, CGGReaT, designed to enhance the presence of minority classes in the anomaly detection domain. Leveraging Large Language Models (LLMs) as a hybrid approach, we use pre-trained transformer-based LLM DistilGPT-2 for generating synthetic tabular data. Extensive experiments on two publicly available benchmark datasets, UNSW NB15 and CIC-IDS2017, underscore the efficacy of our proposed approach. We employed concept drift detection and adaptation techniques to maintain reliable and sustainable ML performance. To enhance interpretability, eXplainable Artificial Intelligence (XAI) methods, including SHAP and LIME, are employed to quantify feature contributions to model outputs. Extensive experiments reveal that testing ML algorithms on datasets balanced with synthetic samples generated by cGGReaT boosts the prediction accuracy on the UNSW NB15 and CIC-IDS2017 datasets, compared to classifiers tested on imbalanced datasets.

Keywords Cybersecurity · Anomaly detection · Data imbalance · LLM · Explainable AI · Concept drift

✉ Iqbal H. Sarker
m.sarker@ecu.edu.au

1 School of Computational and Communication Sciences and Engineering (CoCSE), The Nelson Mandela African Institution of Science and Technology, Arusha, Tanzania

2 Land Management and Valuation, Ardhi University, Dar es Salaam, Tanzania

3 Centre for Securing Digital Futures, Edith Cowan University, Perth, WA 6027, Australia

Published online: 10 February 2026

1 Introduction

The rapid integration of diverse communication devices, combined with advancements in technology and network-based applications, emphasizes the need for intelligent security systems that can detect, monitor, and identify intruders within network infrastructures to maintain confidentiality, integrity, and availability (CIA) [1]. Over the years, Network Intrusion Detection Systems (NIDS) have evolved to counter increasingly dynamic and complex malicious cyber attacks, which can severely compromise network infrastructure. These NIDS are based on ML models. However, in many cybersecurity scenarios, the datasets used to train ML models are highly imbalanced–typically containing a disproportionately large amount of normal traffic data compared to a relatively small number of attack instances. Imbalanced datasets create challenges for accurately detecting intrusions as they result in a bias towards normal traffic, leading to poor model performance on rare attack instances. This rarity of attack representation leads to a minority class in the training data, which adversely affects model learning [2].

Learning from minority class data poses a great challenge to conventional classifiers in the field of supervised learning and data mining. When dealing with imbalanced datasets, the existing traditional classifiers favor majority classes in classification since they were originally intended to process balanced datasets [3]. In using imbalanced datasets in model learning, the prediction ability of the model is highly affected by the imbalanced distribution of the dataset, and the minority instances in the dataset cannot correctly be classified, and sometimes minority classes may not be detected at all [4].

To deal with minority classes, the subject has attracted the attention of many prominent researchers over the years. Researchers have suggested several techniques to improve performance in classifying imbalanced data. The techniques can be divided into three categories: (i) algorithm-level methods, (ii) cost-sensitive methods, and (iii) data-level methods [5]. The algorithm-level technique is focused on modifying or developing algorithms that reinforce the learning towards the minority class [6, 7]. As for the cost-sensitive approach, the approach is designed to minimize the overall cost of incorrect classifications [8]. The data-level technique is the most popular technique to deal with minority classes in intrusion detection datasets. The approach is focused on modifying the data by re-balancing the class distribution and is mainly achieved through over-sampling, under-sampling, or hybrid methods [9–11]. Given this paper, we focus on the resampling technique as the solution to the class imbalance problem.

Several domains within cybersecurity have employed diverse over-sampling and under-sampling methods, such as the Synthetic Minority Over-sampling Technique (SMOTE), Adaptive Synthetic (ADASYN), Random Sampling (RUS), and Near-Miss, to mitigate issues related to data imbalance [12]. Random over-sampling of the minority classes is among the most common and straightforward methods to balance the datasets. In the Random oversampling technique, the minority classes are randomly selected and duplicated to balance the distribution of datasets. The limitation of this approach leads to data over-fitting [13]. An alternative approach to deal with the overfitting problem in the Random sampling technique is by generating the minority class employing an interpolation mechanism based on the nearest neighbor approach [9].

However, state-of-the-art data resampling methods like the Synthetic Minority Oversampling Technique (SMOTE) suffer from the problem of data over-generalization. In such a scenario, generative models play a crucial role in dealing with the situation.

Variations of Generative Adversarial Network (GAN)-based over-sampling techniques have been studied by several researchers for generating synthetic samples to balance the minority and majority classes [14]. Recent technological developments in neural network-based approaches achieve higher quality results compared to classical approaches. GANs can avoid the difficulty of approximating many intractable probabilistic computations to the real approximated distributed instances [15, 16]. GAN and its varieties generate sample instances that are close enough to existing sample instances, which leads to the high performance of ML models.

This research article presents cGGReaT, a data generation approach that combines the advantages of conditional generative adversarial networks (cGAN) with the Generation of Realistic Tabular data (GReaT) method [17], leveraging transformer-based neural networks. We employ generative AI methods in the development of robust synthetic datasets for networks. The aim is to tackle the issue of imbalanced datasets in intrusion detection systems by creating realistic samples of the dataset's minority classes. To prevent bias towards any specific technique, we maintained equal data quantities from both the cGAN and GReaT methods during model training and testing. We applied a conditional GAN for generating 50% of the necessary synthetic attack data. Additionally, GReaT, a generative model based on transformers, was used to create the remaining 50% of data instances required by our cGGReaT approach. This data-generation method included the fine-tuning of a generative pre-trained transformer-based Large Language Model (LLM) (GPT-2) [18] on our tabular dataset, which is then utilized to produce synthetic tabular data. DestilGPT-2, an open-source pre-trained Large Language Model (LLM) by OpenAI [19], functions as a data generator. Furthermore, the GReaT model enables the customization of the GPT-2 LLM to suit our unique datasets, facilitating the creation of synthetic data.

The generated synthetic tabular data is combined with its original data to form the balanced, augmented training data. After balancing the data, four conventional classifiers, including XGBoost, CatBoost, LightGBM, and eXtraTrees, are trained on the augmented dataset. The experiments were conducted on the UNSW NB15 and CIC-IDS 2017 datasets. These datasets are highly imbalanced because the frequency of anomalous data points is frequently underrepresented to that of regular data points. Such a skewed distribution of data may negatively affect anomaly detection algorithms' effectiveness.

We conducted ablation study to find the contribution of each technique towards model performance [20]. Knowing the impact of each component in the model helped improve model design optimization and gain better understand on how the model works [21]. Furthermore, given the dynamic and evolving nature of cyber threats, the statistical properties of network traffic data are likely to change over time, leading to a drift in data distributions [22, 23]. This drift can severely degrade the performance of machine learning models trained on static datasets [24]. We applied the concept drift detection and adaptation [25] to enhance the robustness and practical relevance of the proposed cGGReaT approach.

After performing experimental works for anomaly detection in an Intrusion Detection System (IDS), Explainable Artificial Intelligence (XAI) was used to answer the question of whether we should trust the prediction of the Black-Box models or not. The explainability technique embodied in XAI helps to increase the explainability and transparency of the black-box model predictions by making complex interpretations [26, 27].

Our data augmentation approach addresses class imbalance, enhancing model explainability. Using cGANs and LLMs generates balanced datasets, improving learning from minority classes. Augmentation boosts anomaly detection and response to cyber threats, yielding reliable threat analysis. Combined with XAI techniques, this leads to robust, interpretable insights.

The contribution of our work is summarised as follows:

- A resampling method has been proposed, merging the advantages of conditional generative adversarial networks (cGAN) with the Generation of Realistic Tabular data (GReaT) to tackle data imbalance issues in datasets by utilizing LLMs.
- Fine-tuning the LLM involves using the original tabular dataset to create a synthetic dataset for the representative.
- We analyze five learning models to study the effect of data resampling on them and compare them with each other and with previous works. Those models include XGBoost, CatBoost, LightGBM and EXtra trees. The effect of balanced data representation is observed in all models.
- We integrate concept drift into our proposed model to continuously monitor the model's performance over time.
- Development and description of the model's predictions by applying XAI concepts, and pinpointing the key features as well as the implications of prediction or detection outcomes utilizing LIME and SHAP.

The rest of the paper is organized as follows: Sect. 2 presents related work this research. Section 3 details our proposed generative data augmentation methodology. Section 4 covers experimental results, concept drift analysis, comparative analyses, and an ablation study to evaluate proposed approach effectiveness. Discussion are provided in Sect. 5. Finally, Sect. 6 concludes the research and discusses potential future work directions.

2 Related Works

This section reviews several related works of anomaly detection, explainable AI, and explainable AI-based IDS models.

2.1 Machine Learning for Intrusion

Machine Learning has been widely employed in Cybersecurity for the detection of network intrusions [28–30]. Gu and Lu [31] proposed an effective SVM combined with a naïve Bayes feature embedding approach designed for intrusion detection. Their study reported high accuracy levels, achieving 93.75%, 98.92%, 99.35%, and 98.58% using

the UNSW-NB15, CIC-IDS2017, NSL-KDD, and Kyoto 2006+ datasets, respectively. In [32], researchers evaluated an advanced KNN algorithm integrating a local outlier factor (LOF) on the CICIDS2017 dataset, attaining a 92.74% accuracy in zero-day attack prediction.

Machine Learning based IDSs are accurate and effective at spotting network threats. Nevertheless, high-dimensional data and unbalanced datasets pose a challenge to these systems, leading to reduced accuracy and increased false positive rates. The authors in [33] implemented a filter-based feature reduction method utilizing the XGBoost algorithm on the UNSW-NB15 intrusion detection dataset. The study used the reduced feature space to implement many ML techniques, including Support Vector Machine (SVM), K-Nearest-Neighbours (KNN), Logistic Regression (LR), Neural Network (NN), and Decision Tree (DT). The results show that the XGBoost-based feature selection methodology enables methods such as the DT to improve the binary classification scheme's test accuracy from 88.13% to 90.85%.

2.2 Data Balancing Techniques

Imbalanced data refers to a problem faced by ML models where the number of observations recorded in one class is much lower than those observed in the other classes. Over the years, this problem has existed and researchers have proposed both statistical and ML models to solve this problem. The simplest and most common methods for data balancing are under-sampling [34] and random over-sampling [9]. The drawback of the under-sampling technique is that it tends to discard useful and essential information. In random over-sampling, minority sample instances are duplicated randomly to balance dataset distribution. Random oversampling is prone to overfitting of data [13].

Yanfang et al. [35] presented an ADASYN oversampling algorithm to tackle the data imbalance problem and the stacked autoencoder model as the data downscaling method characterized by a higher dropout structure. A deep learning model for NIDS was suggested for network anomaly detection and integrated a Bi-LSTM. Consecutive characteristics from data traffic were initially extracted via a CNN model, then reconfigured through the attention mechanism of the weights of each channel. Lastly, Bi-LSTM is used to identify the network of sequential features. The proposed model attained an accuracy of 90.73% and an F1 score of 89.65%.

Authors in [36] suggested an Imbalanced Generative Adversarial Network (IGAN) to address the problem of class imbalance. The minority samples were enriched with IGAN to deal with the issue of the low detection rate of minority attacks caused by the imbalance in a dataset. Anomaly detection was achieved through an ensemble of Lenet 5 and Long Short-Term Memory. The proposed model performed above 98% accuracy.

To address the problem of data imbalance in networks, Yang et al. [37] developed a technique known as Self-Paced Ensemble and Auxiliary Classifier Generative Adversarial Networks (SPE-ACGAN). SPE-ACGAN employed a strategy to increase the representation of minority class samples by oversampling using ACGAN while simultaneously reducing the number of majority class samples through undersampling

using SPE. CICIDS2017 and CICIDS2018 were combined to generate a more balanced dataset known as CICIDS-17-18. Experimental results show that SPE-ACGAN outperforms other resampling techniques in terms of performance metrics.

The authors in [16] introduced a data generative model (DGM) to improve the minority class presence in the anomaly detection domain. The approach uses a conditional generative adversarial network to generate synthetic samples for minority classes. To ensure that the generated samples accurately capture the characteristics of the minority class distribution, we incorporated KL-divergence as a guiding mechanism for the model. Publicly available NSL-KDD and UNSW-NB15 datasets demonstrated the proposed approach's effectiveness.

Ahmed et al. [28] developed a model based on the SMOTE data resampling technique. Classification algorithms such as Random Forest, Decision Tree, Logistic Regression, K-Nearest Neighbors, and Artificial Neural Network are used to classify anomaly intrusions using the UNSW NB15 dataset. Results indicated that the Random Forest algorithm achieved the highest accuracy of 89.29% without applying the data resampling method. After applying the SMOTE, the model achieved an accuracy of 95.1%.

In light of successes, transformer-based models have been devised for tabular data classification [38, 39] and learning joint representations of tabular and textual data [40]. Padhi, et al. [41] proposed a transformer architecture based on BERT to study the creation of multivariate time series data. In their analysis [17], the authors generated highly realistic non-sequential tabular data with the help of attention-based LLMs. The proposed technique is capable of modelling tabular data distributions by conditioning on any subset of characteristics, with no extra effort involved in sampling the remaining features.

2.3 Concept Drift

Concept drift refers to a phenomenon in which a model, originally trained on historical data, has less accuracy and efficiency in predictions as the statistical characteristics of a target variable change over time [23, 42]. Continuously monitoring and adjusting model performance is a challenge in data analysis and machine learning [22].

In [43], researchers developed INSOMNIA, an intrusion handling technique that continually updates the underlying ML model when CD happens. Label estimation and active learning are both used to lower labeling overhead and model update latency, respectively. The authors used an explainable AI to understand better how the model responds to changing distributions. Nevertheless, the recurring, blip, and combined variants of CD are not addressed by this method. The authors of [44] presented the Fast Hoeffding Drift Detection Method, which employs Hoeffding's inequality and a sliding window to identify drifts more quickly. Approach for identifying Botnet cyberattacks was proposed in [45], utilizing a dynamic sliding window based on residual projection. The technique employs concept drift analysis and dynamically adjusts the sample size by identifying abnormalities during the process of finding concepts in data streams.

2.4 XAI-Based IDS

Intrusion detection systems (IDS) often emphasize accuracy, but the interpretability of predictive algorithms receives less focus. Many machine learning models are considered "black boxes," which makes them hard to examine. This lack of clarity poses a challenge for security analysts and stakeholders in understanding classification criteria. Explainable Artificial Intelligence (XAI) techniques have gained popularity in addressing this issue, seeking to enhance the interpretability and reliability of ML models, especially in the context of network intrusion detection [27, 46]. Recent research [47] has highlighted the potential and importance of developing XAI techniques in the realms of privacy and security.

To demystify the black-box aspect of sophisticated models, Patil and colleagues [48] presented an ML-based IDS that emphasizes the potential of explainability in security applications. Moreover, Wang et al. [30] and Barnard et al. [49] have significantly advanced understanding and trust among network security experts by integrating techniques such as SHAP (Shapley Additive Explanations) to elucidate the decision-making mechanisms of their intrusion detection models.

In summary, related works in anomaly detection that consider imbalance problems perform significantly better compared to others that do not handle the imbalance problem. Few of the related works have included the explainability analysis component to address the transparency and trustworthiness of ML models. However, to our knowledge, no research has examined the ensemble use of the cGAN and transformer-based technique(GReaT) for data oversampling.

3 Proposed Methodology

In this section, we describe our proposed methodology to solve the prevailing problem of data class imbalance, which potentially hinders anomaly detection performance in ML models. To solve this problem of class imbalance, we propose an ensemble of synthetic data generation techniques consisting of Conditional Generative Adversarial Network (cGAN) and Generative Real-time Transformation (GReaT). Key components and the overall framework for our proposed methodology are detailed in Fig. 1. The main goal is to achieve good detection performance by mitigating the adverse effects of class imbalance on model performance.

Language models have been pivotal in AI and ML, serving a variety of roles, such as in chatbots, content generation, and natural language processing. Among these models, GPT-2 (Generative Pre-trained Transformer 2) is particularly notable due to its impressive capabilities and flexibility. Developed by OpenAI, GPT-2 is a large-scale language model engineered to generate text that closely mimics human writing. It is founded on the Transformer Architecture introduced by Vaswani et al. in 2017 [50], designed to predict the subsequent word in a given sequence, thereby enabling it to produce logical and pertinent text. Leveraging a deep neural network composed of multiple feed-forward layers alongside self-attention mechanisms, GPT-2 is adept at learning intricate patterns within textual data. In this study, the GReaT [17] framework offered a platform for the fine-tuning of GPT-2. The model underwent comprehensive

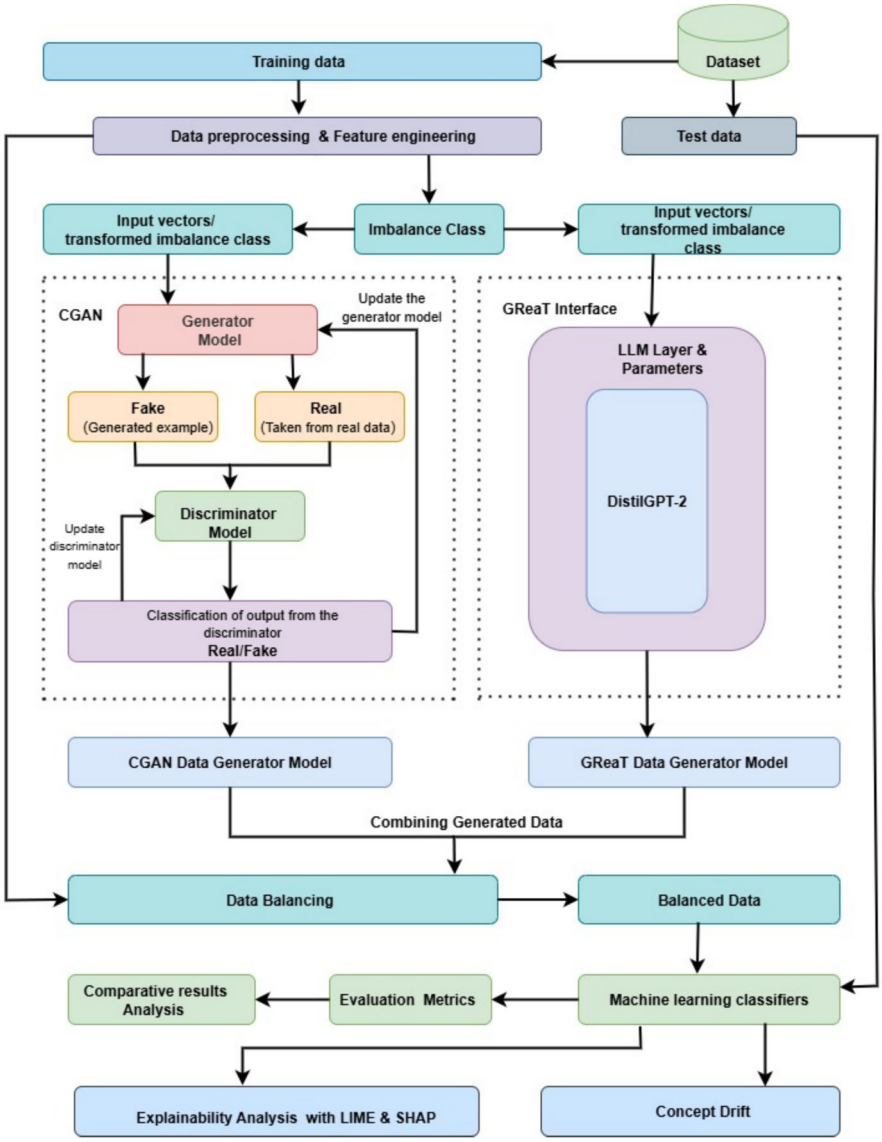


Fig. 1 cGGReaT methodology: a generative AI ensemble approach for minority-class anomaly detection with explainability

fine-tuning to align with our intrusion detection dataset, facilitating the generation of synthetic data.

To evaluate the effectiveness of our data resampling technique, two commonly used intrusion detection datasets were chosen for this research: *(i) CICIDS2017 and (ii) UNSW NB15.* The selected datasets underwent several preprocessing steps before being used for model training and evaluation. These steps include data cleaning, one-

hot encoding, resampling, and normalization to prepare the data for ML security tasks. The datasets were divided into training and test data. The data sampling technique was only applied to the training data, while the resampling process did not touch the test data. The sampled data were applied to individual ML classifiers. Subsequently, models were independently validated using test data and various evaluation metrics, including accuracy, precision, F1-score, and recall. Another evaluation metric applied includes the ROC-AUC score. Finally, we conducted an XAI analysis test to provide a model explanation. We applied specifically local Interpretable Model-agnostic Explanations (LIME) and Global model explainability.

3.1 Dataset Description

3.1.1 UNSW NB15

The UNSW-NB15 dataset is among the most popular evaluated network intrusion detection datasets. This dataset was introduced by the IXIA perfect storm in the Cyber Range Lab at the Australian Centre for Cyber Security (ACCS) [51]. The dataset contains a hybrid of normal activities and attack behaviour. This dataset replaced KDD Cup 1999 and NSL-KDD, addressing several key issues such as obsolete attack types, legitimate traffic scenarios, and the imbalanced dataset in training and testing instances.

The UNSW-NB15 dataset contains nine attack types and one group representing the normal data type. The attack types consist of Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. These classes are explained in Table 1. The original dataset contains 2,540,044 labelled instances stored in four CSV files [51]. Each labelled instance in a dataset is either normal or an attack. The number of records recorded from different types of attacks and normal types for the training set is 175,341 records, and the testing set is 82,332 records, as shown in Table 2. The dataset contains 49 features; however, not all are necessary and relevant for the class label.

Imbalanced classes

In Table 2, the traffic flow of the normal class is significantly higher than the other classes, amounting to 36.09% of the available total dataset instances. Samples of analysis, backdoors, shellcode, and worm classes are much lower than other sample types. The class distribution shows that the worm category has the lowest instances of 130 compared to instances in the training sample of the normal category at the ratio of 1:534. Worms and shellcodes are significantly lower in both the training and test datasets.

3.1.2 CIC-IDS2017

The Canadian Institute for Cybersecurity introduced the CIC-IDS2017 dataset. The dataset comprised benign and the most up-to-date common attacks such as DoS, DDoS, brute force SSH, brute force FTP, heartbleed, infiltration, and botnet, closely

Table 1 Description of UNSW NB15 Network Attacks

Traffic type	Description
Fuzzers	The attacks attempt to crash servers on networks by inputting large amounts of random data
Analysis	An attack targets web applications using web scripts, ports, or emails
Backdoors	Employing a backdoor to secure remote access
DoS	The intruder aims to exploit network resources
Exploits	Target and compromise known vulnerabilities in application or operating systems
Generic	Attacks are related to block-cipher configuration and their keys
Reconnaissance	A probe can circumvent network security controls by gathering relevant information
Shellcode	Code is used to exploit software flaws
Worms	It replicates itself and spreads to other systems through a network computer

Table 2 Class distribution of UNSW NB15 dataset

Class	Training set	Testing set	Total
Normal	56,000	37,000	93,000
Fuzzers	18,184	6,062	24,246
Analysis	2,000	677	2,677
Backdoors	1746	583	2329
Dos	12,264	4,089	16,353
Exploits	33,393	11,132	44,525
Generic	40,000	18,871	58,871
Reconnaissance	10,491	3,496	13,987
Shellcode	1,131	378	1,509
Worms	130	44	174
Total	175,341	82,332	257,673

resembling real-world network flow scenarios [52]. These classes are described in Table 3.

Imbalanced classes

The observed minority classes in the CIC-IDS2017 dataset are Web Attack, Bot, and Infiltration, accounting for only 0.88%, 0.79%, and 0.01%, respectively, of the dataset, which makes the dataset imbalanced. Due to the extensive size of the original dataset, a representative subset of the CIC-IDS2017 dataset was extracted for our experimental analysis. We selected 59,166,40,834 and 100,000 data instances as our representative subset of the CIC-IDS2017 dataset for DoS Hulk, PortScan and Normal classes, respectively.

Table 3 Description of CIC IDS2017 Network Attacks

Traffic type	Description
DoS Hulk	Distributed Denial of Service (DDoS) on a large scale
Port Scan	Detect accessible ports on a network
DDoS	The attacker employs many machines that act together to attack one victim's machine
DoS GoldenEye	The attacker use the GoldenEye tool to launch a denial of service attack
FTP Patator	A brute-force tool is designed to decipher FTP (File Transfer Protocol) passwords
SSH Patator	The brute-force tool is designed to crack Secure Shell (SSH) passwords
DoS Slow Loris	The attacker utilizes the Slow Loris tool to perform a denial of service attack
DoS Slow HTTP Test	Launch HTTP requests designed to be deliberately slow
Botnet	Automated software applications known as bots
Web Attack	Set of malicious actions focusing on web applications
Infiltration	Gaining access to a network without authorization
HeartBleed	A serious security flaw

3.2 Data Preprocessing and Feature Engineering

The datasets, such as UNSW NB15 contain instances with missing and redundant data, which necessitates preprocessing steps to prepare the data for building the classification model. Additionally, UNSW NB15 and CIC-IDS2017 datasets contain nominal features that are not directly compatible with the algorithms used in this study for model training. Therefore, it is essential to convert these categorical features into numerical features. Our study used a label encoder to convert the categorical data to numerical data since GAN and GReaT techniques require numerical input and output variables. Normalizing and standardizing a dataset is a typical prerequisite for ML models.

The Cybersecurity features of the datasets we employed in this study contain a range of values for each feature. A dataset's high numerical feature values significantly impact ML processes on classifiers. Training the high numerical values in a dataset requires a great deal of computational resources. Moreover, when fitting the models with features with large scales, these features dominate others, which leads to misleading results. Our study scaled our data using a standard scaler to normalize the dataset features within normalized ranges [0,1]. Log1p was employed on the UNSW NB15 dataset to reduce skewness.

Feature correlation was used to select features that mostly correlate. In this study, the feature correlation technique was employed to measure the relationship between features and select only those relative features that have moderately higher positive or negative values, i.e., closer to 1 or −1.

Based on a correlation matrix on the UNSW NB15 dataset, features such as sbytes and sloss are highly correlated, and dbytes correlate strongly with dloss. We dropped sbytes and dloss due to high feature correlation. Likewise, features in CIC-IDS2017 were not found to be highly correlated.

Features selection: mutual information strategy

We conducted a feature selection process employing a mutual information strategy [53]. Mutual information determines the statistical dependence between two variables. A higher mutual information score indicates a stronger relationship between the feature and the target, making it more informative for prediction. A low mutual information score denotes a slight reduction in uncertainty between two variables, whereas a zero mutual information score implies that the variables are independent.

Moreover, the mutual information between two variables $X|$ AND $Y|$ denoted $I(X;Y)$, is defined by Cover and Tomas [54] as:

$$\begin{aligned} I(X;Y) &= \sum_{x,y} P_{XY}(x,y) log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)} \\ &= E_{P_{XY}} log \frac{P_{XY}}{P_X P_Y} \end{aligned} \tag{1}$$

Here $P_X(x)$ and $P_Y(y)$ are the marginals: $P_X(x)=\sum_y P_{XY}(x,y)$ (See Table 4).

We calculated mutual information scores for UNSW NB15 and CIC-IDS2017 datasets, please refer to Table 5. To discover the most significant features, a mutual information threshold was calculated using Kernel Density Estimation (KDE) [55], a non-parametric approach for estimating the probability density function of mutual information scores. This technique allows for a more systematic, data-driven identification of essential features rather than depending on arbitrary thresholds or manual selection. The Kernel Density Estimator(KDE) is defined by [55] as:

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{2}$$

Where $\hat{f}(x)$ is the estimated density at point x, n is the number of data points, and x_i is the observed data points. From the KDE equation, h is the bandwidth (smoothing parameter), which controls the width of the kernel function, and $K(\cdot)$ is the kernel function, which determines the shape of the distribution.

The distribution of mutual information scores was visualized through KDE, and a threshold was determined based on the density curve. The determined thresholds were 0.6103 and 0.251375 for the UNSW NB15 and CIC-IDS2017 datasets. Features with scores exceeding these thresholds for each dataset were significantly informative for predicting the target variable. In our case, 38 features from the UNSW NB15 dataset and 65 features from the CIC-IDS2017 dataset were identified as having mutual information values above the thresholds, representing the most relevant predictors. These features were retained, while those with scores below the threshold were excluded for being redundant or weakly correlated with the target.

The threshold-driven approach aligns directly with the use of SelectKBest in feature selection. The k=38 parameter in SelectKBest was chosen based on the KDE analysis on UNSW NB15 and k=65 for CIC-IDS2017, which indicated that precisely 38 and 65 features had mutual information values surpassing the thresholds. By linking the KDE-derived threshold with SelectKBest, the feature selection process ensures that

Table 4 Summary of the selected UNSW NB15 features with mutual information scores

Feature	MI_score	Feature	MI_score
sttl	0.517272095	sloss_log1p	0.302740531
dttl	0.49559592	djit	0.289238547
ct_state_ttl	0.49110569	ct_dst_ltm	0.28195736
rate	0.473080537	ct_srv_dst	0.271736472
smean_log1p	0.464472395	ct_dst_src_ltm	0.270736819
dmean_log1p	0.450940883	ct_src_ltm	0.266493198
sload_log1p	0.448179511	ct_src_dport_ltm	0.264874935
dinpkt_log1p	0.431649122	ct_srv_src	0.262539841
dpkts_log1p	0.429337134	swin	0.249674934
dur	0.422665432	dwin	0.246038497
dload_log1p	0.412047685	proto	0.236136564
synack	0.395825587	service	0.220464942
ackdat	0.394681965	response_body_len_log1p	0.215306729
sinpkt_log1p	0.377664811	is_sm_ips_ports	0.197450844
spkts_log1p	0.344796328	trans_depth	0.191471076
state	0.340658926	ct_flw_http_mthd	0.190688364
sjit	0.322425311	ct_ftp_cmd	0.188392687
dloss_log1p	0.320468055	stcpb_log1p	0.168961377
ct_dst_sport_ltm	0.303648617	dtcpb_log1p	0.163093133

only the most informative features are retained for further modelling. By linking the KDE-derived threshold with SelectKBest, the feature selection process ensures that only the most informative features are retained for further modelling.

Thus we tested our models using 38 selected features for binary and 65 features for multi classification experiments.

3.2.1 Class Balancing

Network intrusion datasets UNSW NB15 and CIC-IDS2017, described above are substantially unbalanced due to the unequal distribution of the various attack categories and the significantly higher number of normal traffic instances than the other attack categories. This problem is known as class imbalance, and in this way, the predictive model created with traditional machine or deep learning methods could be biased and inaccurate [26]. Although the classification accuracy may seem adequate in many situations, there is frequently a noticeable difference between the True Positive Rate (TPR) and False Positive Rate (FPR) values, which suggests that the models have difficulty correctly classifying anomalies. To improve model performances, such as accuracy in imbalanced data, it is essential to balance the dataset using sampling techniques before training the classification models. Several data-balancing techniques have been proposed [9, 16, 56]. In our study, we introduce an over-sampling technique named cGGReaT,and we also compare our proposed approach and other state-of-the-

Table 5 Summary of the selected CIC-IDS2017 features with Mutual information scores

Feature	MI_score	Feature	MI_score
destination_port	0.970893763	flow_duration	0.810319808
total_fwd_packets	0.658734836	total_backward_packets	0.74947025
total_length_of_fwd_packets	0.885558258	total_length_of_bwd_packets	0.952101825
fwd_packet_length_max	0.89247425	fwd_packet_length_min	0.605972028
fwd_packet_length_mean	0.823755641	fwd_packet_length_std	0.637171104
bwd_packet_length_max	0.86211234	bwd_packet_length_min	0.72655274
bwd_packet_length_mean	0.88394183	bwd_packet_length_std	0.675788043
flow_bytes/s	0.779704743	flow_packets/s	0.785273468
flow_iat_mean	0.758594065	flow_iat_std	0.623795152
flow_iat_max	0.80725526	flow_iat_min	0.618122681
fwd_iat_total	0.713719403	fwd_iat_mean	0.699957562
fwd_iat_std	0.603641118	fwd_iat_max	0.753448011
fwd_iat_min	0.558521389	bwd_iat_total	0.581652763
bwd_iat_mean	0.474858379	bwd_iat_std	0.539611174
bwd_iat_max	0.585373222	bwd_iat_min	0.422501374
fwd_urg_flags	0.309062923	fwd_header_length	0.835190029
bwd_header_length	0.889807367	fwd_packets/s	0.767897117
bwd_packets/s	0.864470019	min_packet_length	0.55213978
max_packet_length	0.862305344	packet_length_mean	1.016036419
packet_length_std	0.965615745	packet_length_variance	0.939674428
fin_flag_count	0.348854357	psh_flag_count	0.596675689
ack_flag_count	0.510031365	urg_flag_count	0.427902608
down/up_ratio	0.557847403	average_packet_size	1.00368132
avg_fwd_segment_size	0.823968455	avg_bwd_segment_size	0.883095663
fwd_header_length.1	0.836423507	subflow_fwd_packets	0.658107315
subflow_fwd_bytes	0.885617637	subflow_bwd_packets	0.75044804
subflow_bwd_bytes	0.951696185	init_win_bytes_forward	0.953068237
init_win_bytes_backward	0.829830933	act_data_pkt_fwd	0.569165664
min_seg_size_forward	0.74916842	active_mean	0.628393971
active_std	0.422661736	active_max	0.630225219
active_min	0.625596183	idle_mean	0.625078526
idle_std	0.341970679	idle_max	0.620450373
idle_min	0.625504324		

art sampling techniques, SMOTE [9], Random UnderSampling [57] and Random Oversampling [58].

Random Under Sampling (RUS). One of the easiest ways to deal with class imbalance in datasets is to employ Random Under Sampling (RUS) technique. It randomly selects a subset of data samples from the majority class. After Random undersampling, the dataset's number of instances (of the majority class) decreases, significantly

reducing the training time in a model. However, data points removed by Random undersampling may include important information, decreasing classification results. Subsequently, a randomly chosen subset of the cases from the majority class and the minority class are combined to create a balanced dataset.

Over-Sampling Techniques. We have investigated two popular data oversampling techniques: *(i) Random Oversampling and (ii) SMOTE.* However, this technique produces samples that are similar to the original data, which causes the classification algorithm to become overfit [29, 59].

In 2002, Chawla et al. proposed using the Synthetic Minority Over-sampling Technique (SMOTE) [9] to avoid the over-fitting issue brought on by random oversampling. SMOTE is one of the frequently employed methods to address the class imbalance issue for detecting anomalies. It generates synthetic instances of the minority class to balance its class distribution, thereby avoiding bias and improving model generalization [9]. A linear combination of two similar samples from the minority class creates new minority data points [60]. The minority instance and each of its closest neighbors are uniformly interpolated with new feature values. SMOTE only takes neighboring classes into account. Eventually, a balanced dataset is created by merging the newly created synthetic instances with the original minority instances.

3.2.2 Proposed Data Generation Model

The cGGReaT is the proposed data generative model, which generates synthetic samples using an ensemble of the conditional generative adversarial network (cGAN) and Generative Real n-Time Transformation(GReaT) as shown in Fig. 1

Data generation by GReaT technique: GReaT is a transformer-based language model that utilizes an auto-regressive generating LLM to generate highly realistic synthetic tabular data. This model is notable for its capability to model and create realistic heterogeneous tabular data through a transformer-decoder network architecture. Additionally, the model derives its arbitrary conditioning capability from the GPT-2 Large Language Model, which allows it to model data distributions conditioned on any set of features and to sample the other features. The GReaT methodology centers on generating synthetic tabular data using LLM powered by a pre-trained transformer architecture.

The GReaT model architecture [17] is applied in this research to leverage the use of pre-trained transformer-based LLM DistilGPT-2 [61], a streamlined version of GPT-2, fine-tuned specifically to work with tabular data encoded as text. During fine-tuning, the model is trained to predict the next token in a sequence based on previous tokens, a method known as autoregression [62]. Each row in the dataset is converted into a textual sequence where each feature and its corresponding value are included as part of the text. For example, using the UNSW NB15 dataset, a row with the features state, service, proto, and dur might be encoded as "state: 1.0, service: 0.0, proto: 4.0, dur: 0.121478", with similar encodings for the other features like rate, sttl, and so on.

The sequences are then tokenized using Byte Pair Encoding (BPE) [63], which breaks down the text into subword units like ["state", ":", "1.0", ",", "service", ":", "0.0", ",", "proto", ":", "4.0", ",", "dur", ":", "0.121478"]. The tokenization method effectively balances the granularity of the tokens, capturing both frequent patterns and

rare exceptions in the data. The model then generated text by predicting the next token based on the tokens that had already been generated in autoregressive modelling. The process allows the model to create realistic sequences that mimic the structure and distribution of the original data.

After the model is fine-tuned, it generates synthetic data using weighted choice sampling with a temperature parameter [64] to control the randomness of the generated tokens. Preconditioning techniques allow the model to start text generation from various points in a sequence, providing flexibility in applications such as filling in missing data or generating counterfactual explanations. By preconditioning with specific feature names, the model can produce data distributions consistent with the provided conditions, making it a powerful tool for tasks requiring detailed and context-aware synthetic data generation [65].

Data generation by cGAN technique: cGAN is a neural network that utilizes labels or conditions to produce new text or images with qualities similar to the input it trained on. GANs can overcome the challenge of approximating intractable probabilistic calculations by approximating the distribution of actual training instances. GANs provide synthetic data samples that sufficiently resemble existing samples.

Our proposed network comprises two feed-forward artificial neural networks: the Generator (G) and Discriminator (D). These two neural networks are trained adversarially for data generation as illustrated in Fig. 1. The generator takes input vectors, typically representing features of the imbalance class, and produces synthetic examples that mimic actual data. The generator network G creates these synthetic examples by mapping random noise or conditioned inputs (such as specific class labels) into the data space.

The discriminator network D acts as a classifier, distinguishing between real data from the original dataset and the fake data generated by G. The discriminator provides feedback to the generator on how realistic the generated examples are, allowing the generator to improve iteratively. G tries to create data that D cannot distinguish from the real data, while D tries to become better at identifying fake data. This adversarial process continues until G can generate synthetic data that D can no longer reliably distinguish from real data.

Validation of Generated data instances: We conducted validity test experiments on generated data instances using the Kolmogorov-Smirnov test (KS) [66] on UNSW NB15 and CIC-IDS2017 datasets. The KS test compares the distribution of the actual data instances with the generated data to see if two samples are from the same distribution. The generated data from UNSW NB15 achieved a KS test of 0.15. The generated data instances from UNSW NB15 demonstrate a moderate resemblance to the originals, underscoring the technique's efficacy in maintaining the original data's key characteristics. The generated data instances from CIC-IDS2017 obtained a KS test result of 0.084.CIC-IDS2017 generated data has shown a strong resemblance in distribution with the original dataset.

The cGGReaT approach builds upon existing advanced algorithms for creating authentic data samples. Moreover, in the experiments outlined in this paper, the cGGReaT data generation method outperformed traditional data sampling methods like SMOTE, Random Over Sampling, and Random Under Sampling.

The following is the mathematical model, used to generate synthetic attack samples.

cGGReaT Model
Let D be the original dataset with features X and labels Y. The dataset is split into training and validation sets, D_{train} and D_{valid}, respectively.

$$\text{Class distribution} : P(Y = 0) = \frac{\text{count}(Y = 0)}{\text{count}(Y = 1)} \tag{3}$$

Let $G_{\text{GReaT}}(z; \theta_{\text{GReaT}})$ be the generator function of GReaT, where z is the random noise, and θ_{GReaT} are the parameters.

$$\text{Generated data: } X_{\text{GReaT}} = G_{\text{GReaT}}(z; \theta_{\text{GReaT}}) \tag{4}$$

Let $G_{\text{CGAN}}(z; \theta_{\text{CGAN}})$ be the generator function of CGAN.

$$\text{Generated data: } X_{\text{CGAN}} = G_{\text{CGAN}}(z; \theta_{\text{CGAN}}) \tag{5}$$

Combine generated data from both techniques:

$$X_{\text{combined}} = \text{concatenate}(X_{\text{GReaT}}, X_{\text{CGAN}}) \tag{6}$$

Adjust class distribution in the combined dataset:

$$P(Y = 0)_{\text{balanced}} = P(Y = 1)_{\text{balanced}} = \frac{1}{2} \tag{7}$$

Train a ML model, M on the balanced dataset:

$$M : X_{\text{combined}} \rightarrow \hat{y} \tag{8}$$

Evaluate the model on the validation set D_{valid}. Optimize the model parameters based on a chosen metric. Select important features using a feature selection technique.

3.2.3 Classification Algorithms

To analyze the effectiveness of data generated by cGGReaT, we trained with the most popular and commonly used ML classifiers as follows:

The Extremely Randomized Tree classifier (Extra Trees) is a popular ensemble ML technique. It gathers several decision trees without using bootstrapped samples. Extra Trees use the entire dataset to train decision trees, so they randomly select values at which to split nodes [67].

XGBoost, an advanced gradient boosting algorithm, is a tree ensemble method designed for scalable machine learning, focusing on tree boosting with gradient enhancement. Its scalability stems from significant system and algorithmic enhancements [33]. XGBoost employs parallel and distributed computing, accelerating the model training process. Despite this, configuring the XGBoost model typically requires

fine-tuning several parameters, with its classification performance being greatly affected by parameter selection and their application techniques.

The Catboost classifier, another prominent and effective machine learning algorithm, delivers leading-edge results across numerous experiments. Rooted in gradient boosting, it utilizes binary decision trees as its fundamental predictor and is particularly effective in handling categorical feature prediction [68].

LightGBM is the enhanced variant of the Gradient Boosting Decision Tree (GBDT) algorithm. To provide a well-generalized final prediction, it integrates the predictions of several decision trees. The fundamental concept behind LightGBM is to combine several "weak" learners into one "strong" learner [69].

3.2.4 Concept Drift Detection and Adaptation

We applied concept drift into our proposed model to continuously monitor the model's performance over time. The following is mathematical model used to implement the concept drift detection and adaptation in our proposed model.

The system begins with four trained supervised models: XGBoost, ExtraTrees, LightGBM, and CatBoost for M_1, M_2, M_3, and M_4, respectively, which make predictions on incoming data.

Considering nonstationary streaming data of

$$\mathrm{D} = \{x_1, x_2, \ldots, x_t, \ldots\} \tag{9}$$

Each instance $x_t \in \mathbb{R}^n$ is processed independently in real time. For each incoming x_t, the models produce predictions:

$$\hat{y}_t^{(i)} = M_i(x_t) \tag{10}$$

Subsequent to each prediction $\hat{y}_t^{(i)}$, the system evaluates the actual label y_t and calculates the error:

$$e_t^{(i)} = \begin{cases} 1, & \text{if } \hat{y}_t^{(i)} \neq y_t \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

The error signals are transmitted to ADWIN (ADaptive WINdowing) for dynamic performance monitoring.

ADWIN uses a dynamic window $W_t \subseteq \{e_1, e_2, \ldots, e_t\}$ to detect changes in the error distribution.

The Drift is detected when $|\mu_W^{\text{old}} - \mu_W^{\text{new}}| > \epsilon$. μ_W presents the mean errors in two statistically significant sub-windows, and ϵ is obtained using Hoeffding's bound. If e_t crosses a predefined threshold, it might trigger an update to the model.

Upon concept drift detection, the model needs to adjust itself to accommodate the evolving data distribution. The system retrieves recent data from a sliding window S_t of size k:

$$S_t = \{(x_{t-k}, y_{t-k}), \ldots, (x_t, y_t)\} \tag{12}$$

Each model is retrained on the sliding window data, and the updated models are saved and used for subsequent predictions.

4 Experiments and Results

In this section, several experiments were conducted to evaluate the performance of the proposed models with different data resampling techniques such as Random oversampling, Random undersampling, SMOTE, and cGGReaT. Secondly, the performance of the different models is compared with and without data resampling, and they are also compared with each other's performances and with some state-of-the-art models in intrusion detection. The experiments show that our model outperforms state-of-the-art models in anomaly detection.

4.1 Experiment Setup

Python language was used to develop our proposed system. It was run on Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 1.99 GHz processor and 16 GB RAM. Some details of UNSW-NB15 and CIC-IDS2017 datasets are provided before the experimental results are presented. Due to the advantages of these datasets over the older datasets, the datasets were chosen as benchmarks for this proposed system.

4.2 Evaluation Metrics

A confusion matrix is a commonly used tool for assessing supervised machine learning models. It computes the outcomes of samples that are correctly and incorrectly classified for each class, applicable to both binary and multiclass classifications. To evaluate our proposed system, we must examine outcome results based on precision, recall, F score, ROC value, and accuracy. Firstly, True Positive (TP) refers to the count of attack instances correctly classified. Secondly, True Negative (TN) denotes the count of normal instances correctly identified. Thirdly, False Positive (FP) indicates the number of attack instances incorrectly classified as normal. Lastly, False Negative (FN) represents the number of normal instances that were misclassified as attacks.

Accuracy is the ratio of all correctly classified (normal and malicious)samples, that is,

$$Accuracy = \frac{TP + TN}{(TP + FN + FP + TN)} \tag{13}$$

The Detection rate (DR) is the ratio of accurately detected attack samples, that is,

$$DR = \frac{TP}{(TP + FN)} \tag{14}$$

False Alarm Rate (FAR) is the ratio of false alarms over the total number of alarms.

$$FAR = \frac{FP}{(FP + TN)} \tag{15}$$

Table 6 The optimal hyperparameters of cGGReaT On UNSW NB15

CGAN		GReaT	
Hyperparameter	Setting	Hyperparameter	Setting
Epoch	100	Epoch	2
–	–	Large language Model (LLM)	distilgpt2
Dropout	–	- -	–
Optimizer	Adam	Optimizer	Adam
criterion	BCELoss	–	–
Learning Rate	0.0001	Learning Rate	0.00005
Latent Dimension	200	–	
Layers	2	Layers	–
Batch Size	16	Batch Size	4
–	–	Save Steps	2000
Activation	ReLU &	–	–
Function	Sigmoid	–	
Random Noise	–	–	–
–	–	Logging Steps	50
–	–	Learning Rate Scheduler Type	Constant

4.3 Hyper-Parameter Details of Model

As ensuring a satisfactory convergence when training cGGReaT is frequently challenging, we have carried out several tests to adjust cGGReaT's hyper-parameters. Tables 6 and 7 display the calibrated values of the best hyper-parameters in cGGReaT on the UNSW NB15 and CIC-IDS2017 datasets, respectively. The parameters for generating synthetic data are tailored to network data characteristics such as Network-Specific Architecture and training parameters. In Network-Specific Architecture, for example, cGAN generator and discriminator architectures are designed to match network feature dimensionality. The layer sizes (128, 256 nodes) were chosen based on network feature complexity, and lastly, the latent dimension (200) was selected to capture network traffic patterns. A batch size of 16 was chosen for training parameters to optimize network traffic patterns. The learning rate of 0.0001 was tuned for stable convergence with network data. Furthermore, 100 epochs were determined through convergence analysis.

4.4 Experimental Results

The performance of the proposed models using several data resampling techniques, including SMOTE, random oversampling, and random undersampling, was evaluated by a number of experiments involving a number of algorithms such as XGBoost, CatBoost, LightGBM, and Extra Trees. The experiments show that our model outperforms state-of-the-art models in anomaly detection.

Table 7 The optimal hyperparameters of cGGReaT On CIC-IDS2017

CGAN		GReaT	
Hyperparameter	Setting	Hyperparameter	Setting
Epoch	100	Epoch	2
–	–	Large language Model (LLM)	distilgpt2
Dropout	–	–	–
Optimizer	Adam	Optimizer	Adam
criterion	BCELoss	–	–
Learning Rate	0.0001	Learning Rate	0.00005
Latent Dimension	200	–	
Layers	2	Layers	–
Batch Size	16	Batch Size	4
–	–	Save Steps	2000
Activation	ReLU &	–	–
Function	Sigmoid	–	
Random Noise	–	–	–
–	–	Logging Steps	50
–	–	Learning Rate Scheduler Type	Constant

4.4.1 UNSW-NB15 Experiment

We ran several tests over UNSW-NB15 to evaluate cGGReaT on modern network traffic. To balance its presentation, we generated significant samples for minor classes. In this binary classification experiment, we up-sampled the attack class with 63,341 samples to balance its presentation with that of the normal class.

Table 8 shows the accuracy, recall, F1-score, and precision for both the original and cGGReaT generated data. Table 8 illustrates the effect of the proposed cGGReaT algorithm on the binary classification. It can be observed that most models have improved in terms of accuracy, precision, recall, and F-score. Among the proposed models, the Extra tree model achieved the highest tested accuracy of 97.10% when subjected to upsampled data. In contrast, the same algorithm scored the highest score of 87.38% when tested with the original dataset.Conversely, XGBoost obtained the lowest accuracy, precision, recall, and F1-score values of approximately 89.70%, 90%, 90%, and 90% on resampled data. In contrast, for original data, the XGBoost algorithm achieved an accuracy of 88.52%, precision of 90%, and Recall and F1-score each with 89% and 88%, respectively.

Figure 2 displays the ROC curve and the respective AUC for each algorithm on the resampled UNSW NB15 dataset, with an average AUC of 0.99. Overall, all algorithms showed remarkable performances on resampled augmented data. Furthermore, the evaluation metrics of each of the ML models, XGBoost, CatBoost, LightGBM, and Extra Trees on sampled and original datasets are shown in Table 8. In Tables 8, 9 and 15, A, P, R, and F stand for accuracy, precision, recall, and F1-score, respectively.

Table 8 Classification performance on UNSW-NB15

Classifier	Original data				cGGReaT			
	A (%)	P (%)	R (%)	F (%)	A (%)	P (%)	R(%)	F (%)
XGB	88.52	90	89	88	89.70	90	90	90
CatB	92.05	93	92	92	96.51	97	97	97
LGBM	92.15	93	92	92	96.46	96	96	96
ExTr	87.38	89	87	86	97.10	97	97	97

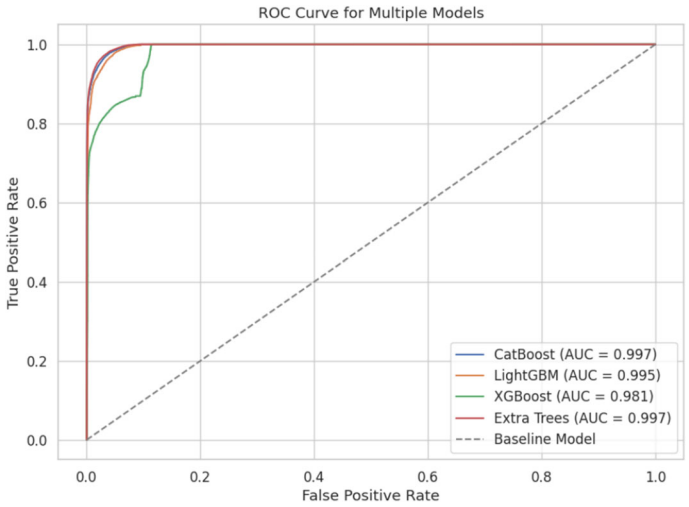


Fig. 2 ROC-AUC curve for multiple models on augmented UNSW NB15 dataset

4.4.2 CIC-IDS2017 Experiment

In this experiment involving augmented data, we upsampled DoS Hulk and PortScan minority classes by 40,834 and 59,166, respectively, while the normal class was kept constant at 100,000 data instances. To evaluate our generated data for minor classes, we trained ML models on both original and augmented data generated by cGGReaT.

Turning to Table 9, we see that ML models on cGGReaT augmented data proved to improve the performance of ML models. Among the proposed models, the Cat Boost model achieves the highest accuracy of 99.73% in the cGGReaT sampled data and 85.13% accuracy on the original dataset.The Light GBM algorithm is observed to have the lowest accuracy of 99.21% and 70.16% for cGGReaT and original data, respectively. Furthermore, the rest of the experimental evaluation metrics of each of the ML, XGBoost, CatBoost, LightGBM, and Extra Trees models in the multi-class classification case on the CIC-IDS2017 dataset is shown in Table 9.

Table 9 Classification performance on CIC-IDS2017

Classifier	Original data				cGGReaT			
A (%)	P (%)	R (%)	F (%)	A (%)	P (%)	R(%)	F (%)	
XGB	70.22	81.00	70.00	59.00	99.27	99.00	99.00	99.00
CatB	85.13	88.00	85.00	84.00	99.73	99.00	99.00	99.00
LGBM	70.16	51.00	70.00	58.00	99.21	99.00	99.00	99.00
ExTr	98.83	99.00	99.00	99.00	99.70	99.00	99.00	99.00

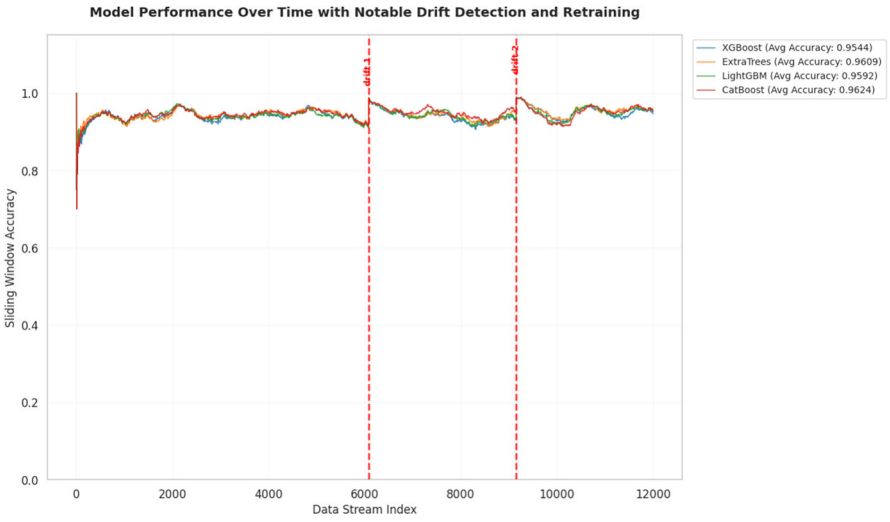


Fig. 3 Training accuracy variation on UNSW NB15

4.5 Concept Drift Analysis

The Figs. 3 and 4 illustrate the training sliding window accuracy of four machine learning models–XGBoost, Extra Trees, LightGBM, and CatBoost–assessed across a continuous data stream of 12,000 instances for each dataset. We used the chunk size of 500 for each dataset, denoting the number of data instances processed in each iteration of the stream. Delta is set to 0.5, representing the drift severity parameter, and accuracy_drop_threshold is set to 0.0001, representing a drop in model performance considered significant enough to trigger a drift warning or alarm. The plots incorporates a drift detection and retraining points, facilitating the evaluation of model efficiency before and after concept drift adaptation. The models in Fig. 3 achieve the average training accuracies of 95.92%, 95.44%, 96.24%, and 96.09% for LightGBM, XGBoost, CatBoost, and ExtraTrees, respectively.

The concept drifts were detected at approximately index 6000 and 9000 using the ADWIN technique.

In Fig. 3, at around data stream index 6000 (drift_1) and index 9000 (drift_2), there are noticeable dips in the sliding window accuracy, indicating changes in data distri-

Table 10 Model validation accuracies for different algorithms on UNSW NB15

Classifier	A%	P%	R%	F%
XGB	88.22	89.00	88.22	88.18
ExTr	95.06	95.11	95.06	95.06
LGBM	94.53	94.53	94.53	94.52
CatB	94.71	94.75	94.71	94.71

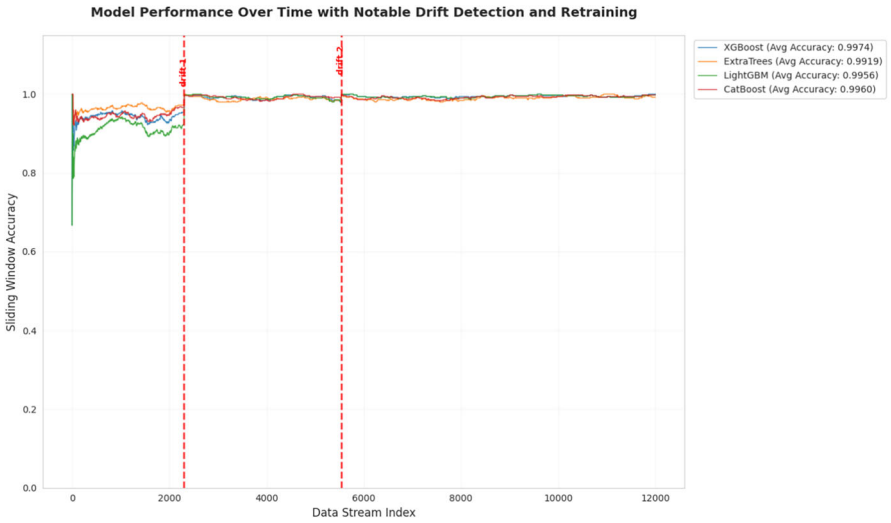


Fig. 4 Training accuracy variation on CIC-IDS2017

bution that lead to a degradation in model performance. Before index 6000, all models illustrate consistent accuracy, suggesting they effectively learned and generalized the initial data distribution. At drift detection points, it is observed that training accuracy temporarily dips slightly, followed by a quick recovery. This shows that retraining or model update occurred promptly after drift detection.

Table 10 shows the validation accuracy of four machine learning models after retraining to address concept drift. This evaluation is critical for determining how effectively each model responds to newly emerging data distributions following a detected drift. Table 10 shows the model's performance after concept drift adaptation, where ExtraTrees, LightGBM, and CatBoost all illustrate strong post-drift validation performance, with accuracies exceeding 94.5%. Likewise, XGBoost shows a notable drop to 88.22%, suggesting it may be less effective at adjusting to the new data distribution.

Likewise, in Fig. 4, the concept drifts were detected at approximately index 2300 and 5600 using the ADWIN technique. There is a noticeable changes (fall) in the sliding window accuracy, signifying changes in data distribution that lead to a model performance degradation. After each identified drift and subsequent retraining, performance rapidly stabilizes, close to baseline levels as shown in Fig. 4.

Table 11 Model validation accuracies for different algorithms on CIC-IDS2017

Classifier	A%	P%	R%	F%
XGB	97.87	97.93	97.87	97.87
ExTr	99.00	99.00	99.00	99.00
LGBM	97.85	97.90	97.85	97.84
CatB	97.80	97.81	97.80	97.80

Table 12 A comparison of weighted F1-scores for UNSWNB15

Model	XGB	CatB	LGBM	ExTr
Original	93	92	92	93
Smote	94	93	93	96
RUS	94	95	95	95
ROS	95	95	95	97
cGGReaT	90	97	96	97

Table 11 shows the validation accuracy of four machine learning models after retraining to address concept drift. The model's performance after concept drift adaptation shows that XGBoost, ExtraTrees, LightGBM, and CatBoost all demonstrate strong post-drift validation performance, with accuracies exceeding 97.80%, as shown in Table 11.

In a real network environment, network attacks evolve, leading to models trained on older data underperforming. A static Intrusion Detection System (trained once and deployed) is inadequate. An adaptive IDS that integrates a concept drift handling mechanism is crucial for ensuring consistently high detection rates and model reliability.

4.6 Comparizon Analysis

To validate the effectiveness of cGGReaT, we implemented data over-sampling and under-sampling techniques to handle data imbalanced problems in network intrusion detection. Different techniques for under-sampling techniques, such as Smote [9] and Random undersampling, were implemented. Conversely, the over-sampling method, Random oversampling, was also implemented to validate our data resampling method. Both resampling methods yield varying results in the classification of network intrusion detection datasets. When compared to existing techniques, the proposed cGGReaT in general, helps to minimize the problem of data imbalances faced by ML models. A comprehensive comparative analysis for both benchmark datasets is illustrated from Table 8 to Table 14.

Table 8 depicts the precision, recall, and F1-score of four classifiers on the binary classification problem. The results in Table 8 show the benefit of using cGGReaT to improve the ML classifiers' performances in intrusion detection when training in highly imbalanced datasets. The cGGReaT technique enhances intrusion detection training dataset quality by generating synthetic data. As a result, classifiers' effec-

Table 13 A comparison of weighted F1-scores for CIC-IDS2017

Model	XGB	CatB	LGBM	ExTr
Original	59.00	84.00	58.00	99.00
Smote	96.00	97.00	88.00	99.00
RUS	91.00	96.00	79.00	99.96
ROS	96.00	96.00	85.00	99.00
cGGReaT	99.00	99.00	99.00	99.00

tiveness improved when trained on the augmented datasets of cGGReaT compared to the original datasets. For example, Extra Trees classifier trained with augmented data attained improved performance approximately by 9% compared to the original dataset. Likewise, Catboost and Light GBM classifiers achieved improved performances of approximately 4% to each with augmented data, in contrast to the XGBoost classifier, which attained an improved performance of approximately 1% compared to the classifier's performance with the original dataset. Table 8 shows that the cGGReaT approach significantly enhances classification performance across all evaluated algorithms, with especially remarkable improvements for the Extra Trees classifier.

The results in Table 9 show that cGGReaT obtained the best performance in all performance measures in the multi-classification problem on CIC-IDS2017. In general, there is a significant improvement in all models trained with the augmented cGGReaT dataset, with the exceptional Extra Trees model, which attained a slight improvement of approximately 1% in all performance metrics; this could be because it is efficient in handling datasets with a large number of features and noisy data.

Conversely, Table 12 compares the F1 scores of different data sampling techniques, SMOTE, ROS, RUS and cGGReaT and on original data. After applying resampling techniques, the F1 scores increased for all algorithms in all resampling techniques compared to the original data. Table 12 demonstrates that the F1 scores of the classifiers trained on the datasets of cGGReaT are notably higher than most of those trained on the datasets of the other resampling techniques. Furthermore, cGGReaT's improvement over the original data is consistently more remarkable than the improvement of the other sampling strategies by a more significant margin. This remarkable improvement confirms that resampling techniques positively impact the performance of most classical ML algorithms.

Likewise, Table 13 showcases that the F1 score of our proposed data generation method on the CIC-IDS2017 dataset attained remarkable performances in most of the classifiers. When the proposed method is compared with the performances of other sampling methods, it is often the highest value among all tested techniques except for the Extra Trees classifier in the Random Oversampling technique. This shows that the proposed data resampling technique is robust in solving data imbalance problems in intrusion detection. However it is noted that F1_score for all resampling techniques is approximately constant for the Extra Trees algorithm since It handles imbalanced data naturally, reducing the need for data resampling.

Table 14 Comparison between proposed and existing works for network anomaly detection

References	Dataset	Approach	Acc%	F1score%	Concept drift	XAI
[16]	UNSW NB15	DGM-SPOCU	–	75.00	×	×
	NSL-KDD	DGM-RELU	–	73.00	×	×
[28]	UNSW NB15	SMOTE	95.10	95.10	×	×
[71]	NSL- KDD	SMOTE	99.64	98.38	×	×
	CIC-IDS2017	SMOTE	99.89	94.78	×	×
Our Study	UNSW NB15	cGGReaT	97.10	97.00	✓	✓
	CIC-IDS2017	cGGReaT	99.73	99.00	✓	✓

In a comparative analysis of prior studies of various scholars such as [16, 28, 70], and [71] that focused on different resampling techniques employing different datasets including ours, are represented in Table 14.

Notably, our suggested data sampling technique, which uses pre-processed sampled data acquired through several pre-processing procedures, including feature selection, shows notable improvements in almost all evaluations, including accuracy and F1 score. However, there are situations when accuracy does not correlate well with true positives and true negatives. In Table 14, we have presented accuracy alongside the F1-Score. From Table 14, the proposed technique performances show that the F1 score has the same pattern as the accuracy, which means that the ability to detect true positives and true negatives is relatively good for our model. Notable differences between our study and the others are the inclusion of experimental explainability analysis tests, which show each feature's contribution to model prediction, and the inclusion of concept drift analysis, which demonstrates how the static data trained model's performance behaves under changing data distributions.

4.7 Ablation Study

Ablation study involves the removal of one or more elements from a ML model or training dataset. The main goal in ablation studies is to demonstrate how a particular component of the system affects relevant models' metrics (e.g. accuracy). Ablation studies are important in both ML and deep learning research, as they are crucial in validating proposed neural network or machine learning models and datasets. To evaluate the contribution of each component in our proposed model on UNSW NB15 dataset, we conducted abrasion experiments on cGAN and GReaT data generation techniques.

Table 15 presents two experimental results obtained when the proposed model tested using the dataset generated by cGan and GReaT methods, respectively. We noted that from the results, the two experiments of cGAN and GReaT data generated techniques did not outperform the baseline of our proposed model when each data generation technique was tested against our proposed model. Ablation results demonstrate that cGAN and GReaT (GPT-based) models offer complementary strengths in synthetic data generation. cGAN model excels at capturing numerical feature rela-

Table 15 Ablation study on the components of our method on UNSW NB15

Classifier	cGAN				GReaT			
	A (%)	P (%)	R (%)	F (%)	A (%)	P (%)	R(%)	F (%)
XGB	80.03	86.00	80.00	79.00	89.23	89.00	89.00	89.00
CatB	88.45	91.00	88.00	84.00	96.94	97.00	97.00	97.00
LGBM	67.65	80.00	68.00	64.00	96.83	97.00	97.00	97.00
ExTr	83.30	87.00	83.00	83.00	96.45	96.00	96.00	96.00

tionships and maintaining statistical distributions, as evidenced by our KS test results, which are shown in the validation of generated data instances section. The GReaT model performs better in handling categorical network features (e.g., status, service, proto), offering more diversified and realistic categorical value combinations.

Table 8 shows that our proposed technique (cGGReaT), combining the strength from cGAN and GReaT methods, outperforms others in terms of fidelity on the dataset. The results reflect the success and effectiveness of our ensemble approach to our proposed model, where both components help enhance the proposed model's performances.

4.8 SHAP-LIME Based Explainability Analysis

The data science community has become increasingly interested in explainable AI in recent years. Unlocking the mysteries of black box models is a popular area of research, with a growing number of tools and libraries being released daily. Thus, we have employed SHAP and LIME explainability techniques to explain the ML model in this study.

4.8.1 SHAP-Based Exlanibility Analysis

SHAP (Shapley Additive exPlanations) uses game theory to explain the features influencing a particular choice. SHAP is a useful method for explaining ML models' predictions. It gives us a means of determining how each feature contributed to a certain prediction, assisting us in comprehending the reasoning behind a model's choice. Both global and local interpretations can apply SHAP to unlock the mysteries of black box models. Using the SHAP Tree Explainer method, this study computed SHAP values to explain the model's behavior for individual instances and the entire dataset. In the SHAP summary global plots for binary classification in Fig. 5a, the red color illustrates the average influence of SHAP values on identifying anomalous attacks. In contrast, the blue color SHAP values indicate typical normal instances. Likewise, in Fig. 5b, the SHAP global summary plot for the multi-classification problem in the CIC-IDS2017 dataset, the blue color SHAP values indicate normal class, the pink color indicates SHAP values on identifying DoS attack class, and the mint color represents the average influence of SHAP values on identifying PortScan attack.

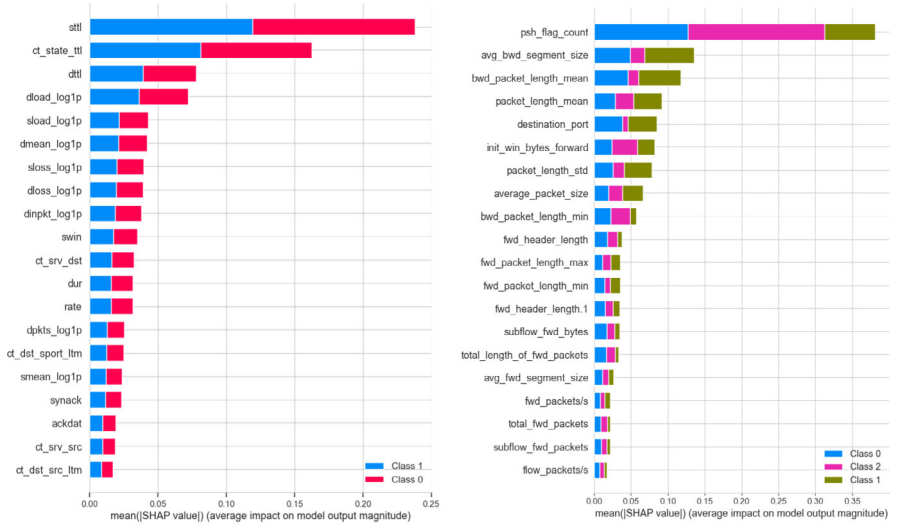


Fig. 5 The summary in hierarchy of the features contributing to classification on **a** UNSW NB15 and **b** CIC-IDS2017

Figure 5 shows a hierarchical feature summary arranged according to how much each feature contributed to the model's output for the UNSW NB15 and CIC-IDS2017 datasets, respectively. The magnitude of the SHAP values indicates the strength of the features, and features with larger absolute SHAP values have a more significant impact on the models' predictions.

Notably, Fig. 5a shows the most influential UNSW NB15 dataset features in the model's predictions. One noticeable feature with the highest score is 'sttl', which identifies potential malicious network traffic by analyzing the expected time-to-live (TTL) values for different network connections. Furthermore, the second-highest mean SHAP value feature is shown to be 'ct_state_ttl', which contributes significantly to anomaly detection. The "ct_state_ttl" represents the "connection state time-to-live" value, specifying how long a connection can remain active depending on its current state inside a network communication flow.

The feature 'dttl' is the third most significant contributor to the anomaly detection. It is time to live, the value of packets sent by the destination host back to the source. Abnormal patterns can reveal attacks from unusual networks or spoofed IP ranges. Similarly, 'dload' represents the total number of bytes sent from the source host to the destination host during a network connection. Unusual patterns of outbound data signify that it is the malicious network traffic flow. The rest of the features are shown in Fig. 5a.

In Fig. 5b, 'flow_packet/s' shows the least contribution to the model's classification. It indicates the number of packets per second within a flow, which helps identify volumetric attacks, behavioral anomalies, and resource-based threats. Specifically, 'psh_flag_count' is the most important, with the most considerable average contribu-

tion. psh_flag_count shows the number of TCP packets within the flow/connection with the PSH (Push) flag set. 'psh_flag_count' indicates the number of TCP packets within the flow/connection with the PSH (Push) flag set. The presence of high patterns in PSH flag usage indicates potential attacks, such as DoS, scanning, or malicious tunneling.

Furthermore, in Fig. 5b, feature 'avg_bwd_segment_size' indicates the average size of packets transmitted backward inside a network flow. Deviations from regular patterns of the feature signal potential attack activities, such as scanning, brute force, and DoS/DDoS. Likewise, the feature 'bwd_packet_length_mean' represents the average packet length in the backward direction. Abnormal values of the feature may indicate the presence of scanning, brute force, DoS/DDoS, or data exfiltration attack activities. The rest of CIC-IDS2017 features contributing toward the models' predictions is shown in Fig. 5b.

Studying the overlap or divergence of important features between the two datasets in Fig. 5 could provide valuable insights into generalization and model transferability. The UNSW-NB15 and CIC-IDS2017 datasets employ different feature engineering pipelines; however, a set of conceptually equivalent or similar features is present in both datasets, as shown in Table 16. These feature overlaps are vital for cross-dataset generalization. Table 16 shows that even though the feature naming is different, both datasets' top features may capture similar traffic characteristics.

For example, features such as 'smean' of UNSW NB15 and CIC-IDS2017 features 'avg_fwd_segment_size' and' fwd packet length mean' all reflect the average packet size in the forward direction of a network flow, which helps to detect anomalies like DoS, DDoS, and portscan activities. Likewise, features like 'rate' found in UNSW NB15 and CIC-IDS2017 feature 'flow packets/s' all represent the number of packets per second within a given flow, which could help capture how frequently packets are sent in a flow, making it a crucial feature for detecting anomalous network flow behaviors such as DoS and DDoS. Furthermore, the feature 'dmean' from UNSW NB15 and the features 'avg bwd segment size' and 'bwd packet length mean' of the CIC-IDS2017 datasets are conceptually very similar. All three features in common measure the mean size of data segments transferred in the reverse direction of a network flow. Specific anomalous patterns detected by these features include port scanning activities, DDoS/DoS attacks, and data exfiltration. The rest of the conceptually similar features that cut across datasets are listed in Table 16.

Conversely, top features of each dataset in Fig. 5 show divergence with features of another dataset in terms of network traffic characteristics. The UNSW NB15 top predictors such as 'sttl','ct_state_ttl', dttl are packet-level behavioral features which capture TTL irregularities, exposing spoofing/evasion attempts, and providing flow state context for better attack differentiation. In CIC-IDS2017 top features in prediction output such as 'psh_flag_count','packet_length_std', 'bwd_packet_length_min' and 'fwd_packet_length_max' focus on packet-level statistics (header length, max packet size). Thus a model trained solely on one dataset might fail to capture these other traffic patterns of another dataset.

SHAP calculates the Shapley value to show the influence of features on the model predictions. We computed the SHAP values for a specific instance that we chose.

Table 16 Conceptual Similarity of selected features for UNSW NB15 and CIC-IDS2017

UNSW-NB15 Feature	CIC-IDS2017 Feature	Conceptual Similarity
smean	avg_fwd_segment_size fwd_packet_length_mean	Average packet size in forward direction
dmean	avg_bwd_segment_size bwd_packet_length_mean	Average packet size in backward direction
sload	subflow_fwd_bytes total_length_of_fwd_packets	Volume of data sent forward
dload	subflow_bwd_bytes total_length_of_bwd_packets	Volume of data sent backward
rate	flow_packets/s	Overall packet rate
ct_srv_dst	destination_port (indirect)	targeting of same service/port repeatedly
ct_dst_sport_ltm	destination_port + flow count	Service/port-level activity concentration
dpkts_log1p	total_bwd_packets	Total packets in backward direction
swin	init_win_bytes_forward	Initial TCP window size forward

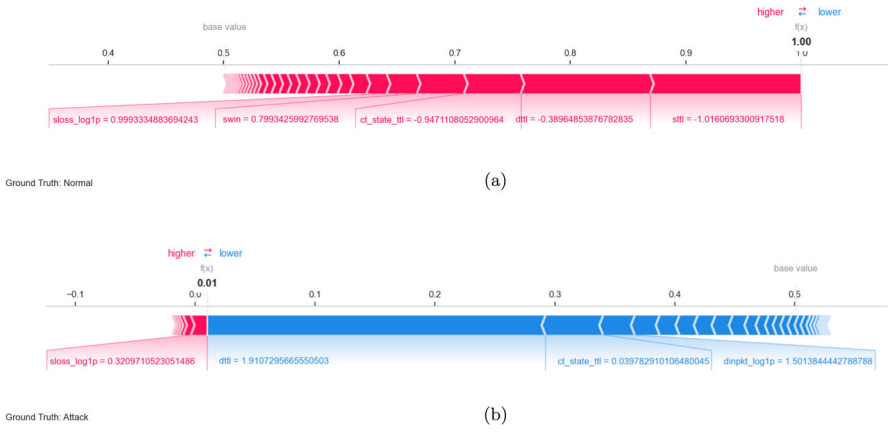


Fig. 6 The local plots of instances on the UNSW NB15 dataset show each feature's contribution to the prediction,(a) Normal class instance, (b) Attack class instance

Figure 6 shows the force plot of the 'normal' and attack instances, showing each feature's contribution towards model prediction. The average of all the predicted values constitutes the plot's base value. Every plot strip displays how the features affect the prediction value. Each strip in the plot shows the impact of the features in pushing the predicted value closer or farther from the base value. The plot in Fig. 6a shows a local explanation where the probability of the normal instance output is 1.00, the features 'sloss_log1p', 'swin', 'dttl','ct_state_ttl' and 'sttl' are the most contributed features, which makes the instance to be classified as non-attack.

Figure 6b shows the force plot of the instance UNSW-NB 15 test sample labeled attack. The model's prediction for this instance is shown in bold, i.e., 0.01, which is less than the base value. It is observed that blue-colored features dttl, 'ct_state_ttl'

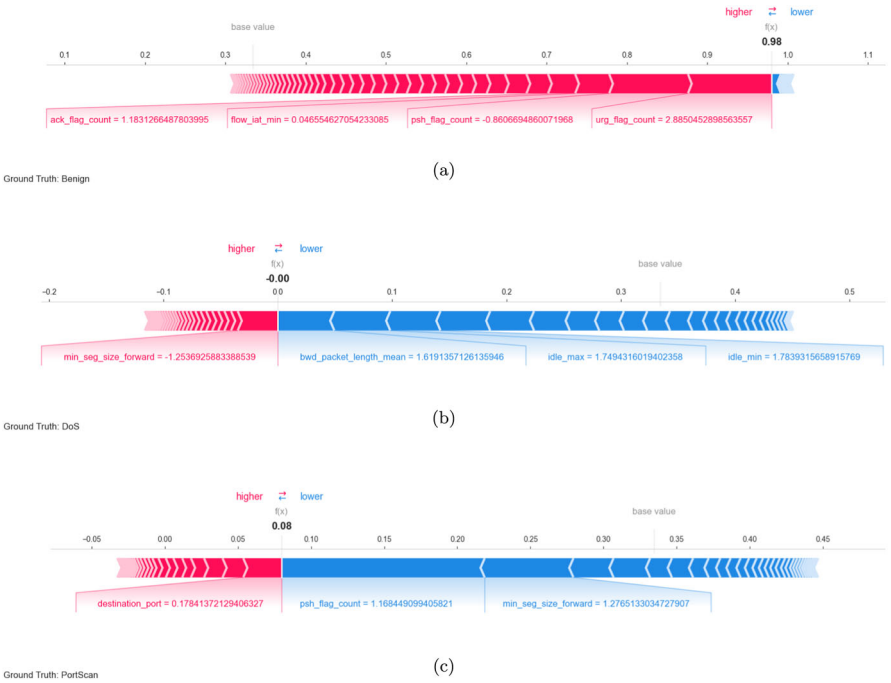


Fig. 7 The local plots of instances on the CIC-IDS2017 dataset show each feature's contribution to the prediction, **a** Normal class instance, **b** Dos attack instance, **c** Portscan attack instance

and 'dinpkt_log1p', are forcing the classification towards lower, i.e., attack while sloss_log1p drive prediction higher to make the classification as an attack.

Figure 7 shows force plots experimented on CIC-IDS2017 dataset, illustrating the effect of SHAP values on the interaction of features and the overall prediction at the individual level. The sum of the effects of each feature value in conjunction with the base value influences the model's output prediction. The base value is displayed in the plot and the features that positively influence the predictions are highlighted in red. In contrast, the features with a negative impact are highlighted in blue. The average of each predicted value serves as the plot's base value. While blue strip features push the value to lower values, red strip features push the value to higher values. Features with wider strips contribute more.

Figure 7a, provides a local explanation where for the specific data instance chosen from the CIC-IDS2017 dataset, the base value is 0.334 and the model's predicted value is 0.98. All features, 'ack_flag_count', 'flow_iat_min','psh_flag_count' and 'urg_flag_count', have positive contributions to the prediction value. 'urg_flag_count' feature is the most significant feature, given that the contribution has a broader range. Given that the final predicted value exceeds the base value, the predicted class is Normal.

Similarly, we selected instances from the same CIC-IDS2017 dataset, as shown in Fig. 7b, where the base value is 0.33 and the prediction output is 0.00. The predicted class for this instance is a denial of services (DoS). From the plot,

'bwd_packet_length_mean', 'idle_max' and 'idle_min' features push prediction values to lower, meanwhile 'min_seg_size_forward' pushes the model's prediction to the higher value. For instance, the 'min_seg_size_forward' feature denotes the minimum-sized segment observed in the forward direction of a network flow, which shows an attack indicator, making the model lean towards a DoS attack classification.

In Fig. 7c, '$destination\_port'$ increases prediction value $f(x)$. The plot shows a data instance classified as Portscan. The 'destination_port' feature is a strong indicator of a scanning attack. Attackers frequently probe numerous ports to identify open services, triggering portscan detections.

In addition to improving performance metrics, applying the LIME technique enhanced the interoperability of the IDS model's predictions. We could determine explanations for particular occurrences using the LIME technique, which gave us insight into the factors influencing the model's decision-making. The LIME explanations made it possible to understand the specific features and their weights that went into the model's predictions. For instances of attacks, LIME explanations highlight particular network traffic attributes such as a lengthy connection time, a high number of unsuccessful login attempts, and a large volume of data transfer as determinants leading to the model's choice to categorize an instance as an attack. These explanations help analysts understand the underlying reasons for the predictions made by the IDS model and provide transparency and interoperability critical in essential decision-making processes.

In this regard, LIME has also been employed in this research to explore the model's choices for specific predictions [72] in binary classification using UNSW NB15 and multi-classification using CIC-IDS2017 datasets.

Figure 8b explains how to establish if the classification result is 'Normal' or 'Attack', along with probability and original instance values. Colors indicate which features contribute to which class. Orange-colored features indicate the 'attack' class, whereas blue-colored features indicate the 'normal' class. The LIME results consist of three pieces of information from left to right. The information includes the model's predictions, the features' contributions, and the actual value of each feature. In Fig. 8a and b, we visualize LIME explanations for our model's predictions. The predicted value is benign with 100% accuracy, as shown in Fig. 8a, with 'sttl','dtll','ct_flw_http_mthd','trans_depth','stcpb', 'ct_dst_src_ltm', 'ct_src_dport_ltm' and 'swin' the most influential features affecting the model's decision. Conversely, another data record instance chosen to form a testing sample shows that the probability value is an attack with an accuracy of 100%, as shown in Fig. 8b.

Figure 9 represents LIME explanations for three different instances from the CIC-IDS2017 dataset. Figure 9a highlights the most influential features leading to the model's decision. Notably, features such as 'destination_port', 'psh_flag_count', 'idle_max', 'idle_min', 'idle_mean', 'fin_flag_count', 'bwd_packet_length_min', 'fwd_iat_max' and 'bwd_packet_length_std' with the assigned weights of 0.04, 0, 03, 0.03, 0.02, 0.02, 0.01, 0,01, 0.01 and 0.01 respectively contribute to the model's accurate prediction. The predicted value is aligned with benign with 99% accuracy, as shown in Fig. 9a. The 'fwd_length' feature, with the weight of 0.02, pulls classification away from benign, but the assigned weights from features contributing to

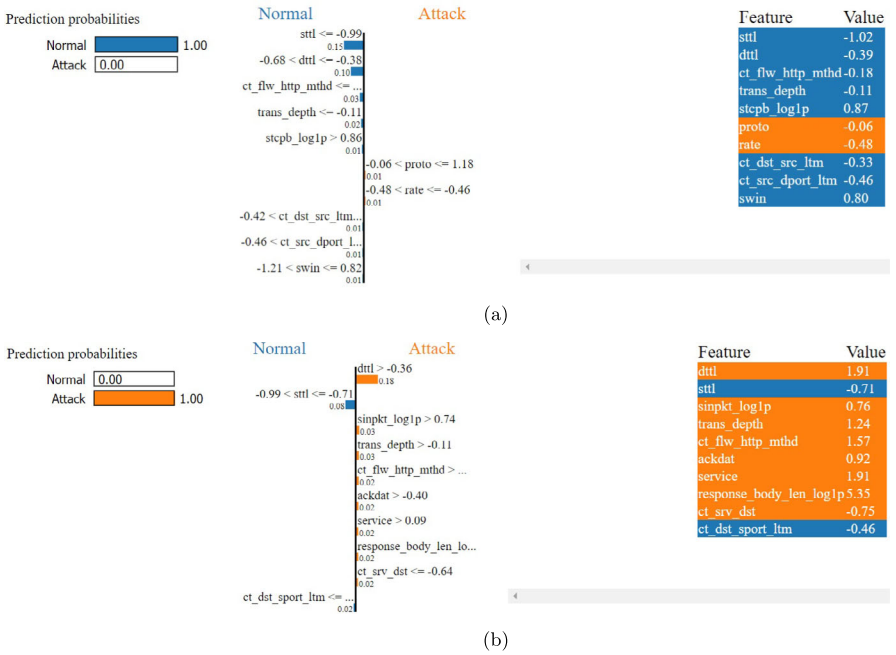
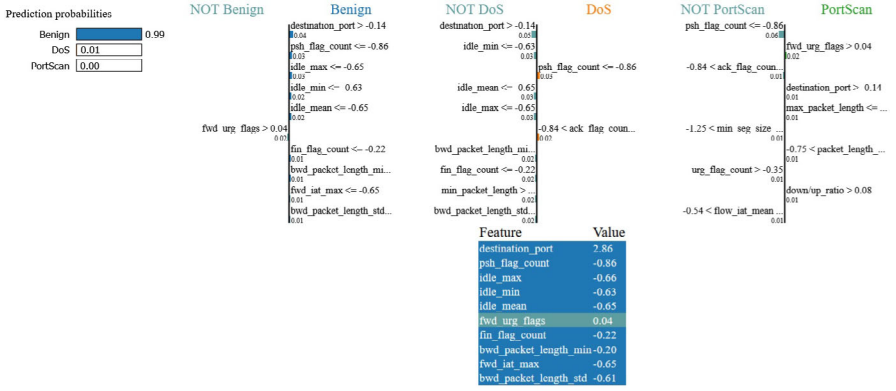


Fig. 8 Explanation of the LIME observations for **a** normal class instance, **b** attack class instance

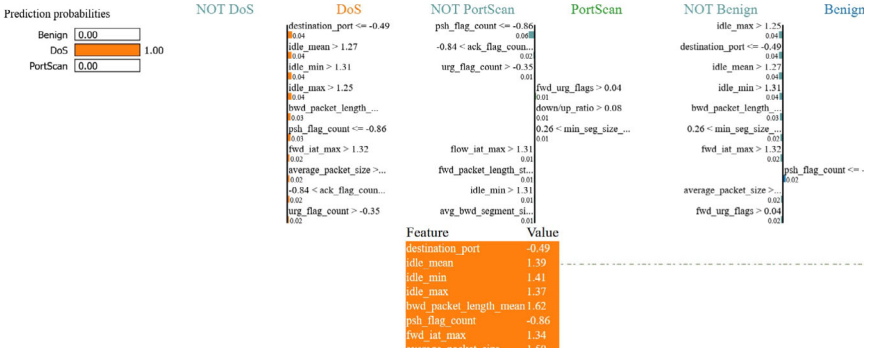
benign overcome its weights. These features suggest that high idle times, low push flag counts, and low PortScan activities characterize normal traffic in our model.

Similarly, the model predicted this instance as a DoS attack with 100% accuracy, as shown in Fig. 9b; the Figure illustrates feature values that led to the correct classification of a DoS attack. The probabilities for Benign (0.00) and PortScan (0.00) are negligible, indicating that the feature values are consistent with DoS attack behavior. Features with their respective weights like 'destination_port:0.04', 'idle_mean:0.04', 'idle_min:0.04' 'idle_max:0.04', 'bwd_packet_length_mean:0.03', 'psh_flag_count:0.03','fwd_iat_max:0.02', 'average_packet_size:0.02','ack_flag_count:0.02' and 'urg_flag_count:0.02' have significantly influence in detecting DoS attack. Characteristics like high idle time (Idle Time (idle_max, idle_mean, idle_min) suggest intermittent DoS flooding behavior. The presence of low psh flag counts further confirms DoS attack traffic.

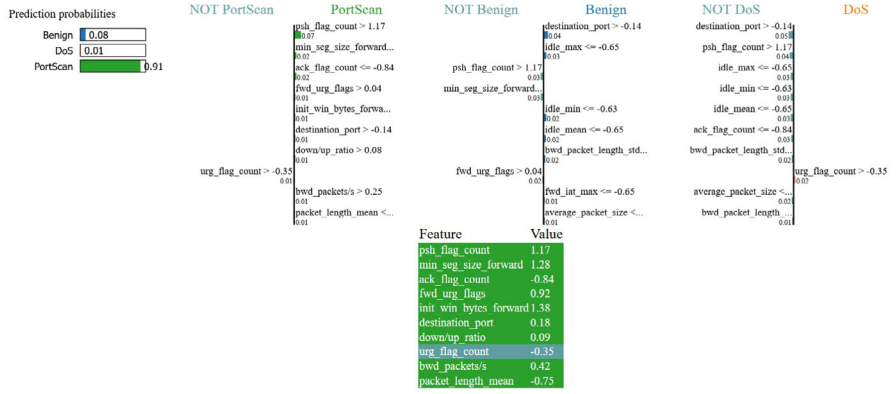
Lastly, Fig. 9c displays the prediction of a Portscan attack, with a likelihood of 91%. This figure highlights the features influencing the model's predictions. The feature most strongly supporting the prediction is '$psh\_flag\_count > 1.17'$, with a weight of 0.07. Following this, $'min\_seg\_size\_forward > 1.28'$ and $'ack\_flag\_count <= -0.84'$, each with a weight of 0.02, rank second in significance for predicting a Portscan attack. Other contributing features, each carrying a weight of 0.01, include $'fwd\_urg\_flags > 0.04'$, $'init\_win\_bytes\_forward > 1.38'$, $'destination\_port > -0.14'$, and $'down/up\_ratio > 0.08'$. A high '$psh\_flag\_count > 1.17$', a low '$ack\_flag\_count <= -0.84$', and a high



(a)



(b)



(c)

Fig. 9 Explanation of the LIME observations for **a** Normal class instance, **b** Dos attack instance, **c** Portscan attack instance

‘$down/up\_ratio > 0.08$‘ substantiate the model’s prediction of a portscan attack. Furthermore, the cumulative feature weights contribute positively to the model’s decision.

These LIME explanations provided insight into features’ contributions that influence our models’ predictions.

5 Discussion

This section reviews the experimental results, highlighting the cGGReaT approach contributions, strengths, practical applications, interpretability, and limitations.

5.1 Effectiveness of cGGReaT for Anomaly Detection

The cGGReaT method effectively addresses data imbalance, significantly boosting classification metrics across anomaly detection algorithms. Notably, in Table 8, the Extra Trees algorithm accuracy improved from 87.38% to 97.10% on UNSW-NB15, and the Catboost classifier in the Table 9 increased its accuracy from 85.13% to 99.73% on the CIC-IDS2017 dataset. Integrating conditional GANs for numerical data and GReaT for categorical features provides a robust approach to synthetic data generation for balanced anomaly detection.

Experiments showed that classifier algorithms perform well with imbalanced data, achieving 98.83% accuracy on the original dataset (Table 9). Our technique improved performance by 1%. Our evaluation shows cGGReaT outperforms traditional methods like ROS, RUS, and SMOTE. Combining cGAN with transformer based techniques consistently beats these benchmarks.

Our proposed model was tested and evaluated in a concept drift environment using the UNSW NB15 and the CIC-IDS2017 datasets. The evaluation of the proposed model effectively handles concept drift, although its performance varies with the dataset characteristics. In UNSW NB15, accuracies dropped slightly after drift, particularly for XGBoost, demonstrating its susceptibility to changes in data distribution. Conversely, Extra Trees, LightGBM, and CatBoost demonstrated increased resilience, sustaining more consistent outcomes. However, all models consistently achieved high validated accuracies above 97% on CIC-IDS2017, with Extra Trees achieving 99.00%, highlighting the proposed approach’s robustness in handling drift. Although the proposed model faced with drift severity and data distribution changes impacting its behavior, Extra Trees emerging as the most drift-tolerant across datasets due to its randomized splits and ensemble diversity.

Despite often being ‘Black Boxes’, machine learning models can benefit from XAI techniques. Explainable AI techniques SHAP and LIME improved model transparency, as detailed in section 4.7. SHAP offered global interpretability by showing feature influence on decisions, while LIME provided local interpretability, helping analysts understand specific predictions. These improvements are vital in Cybersecurity, enhancing user trust by clarifying prediction rationales and boosting decision-making confidence. They also improved anomaly detection by analyzing feature relevance;

for example, 'sttl' indicates packet lifespan, 'Psh_flag_count' signals scanning activities, and 'destination_port' identifies targeted services. Understanding these features boosts classifier accuracy and intrusion detection effectiveness.

5.2 Practical Implications

The ablation studies revealed critical insights about the complementary strengths of cGAN and GReaT. cGAN demonstrated remarkable efficiency in capturing numerical feature distributions, as evidenced by successful synthetic data generation. Conversely, GReaT was particularly effective in managing categorical variables with context-aware synthesis. The integrated cGGReaT model leveraged these complementary strengths, resulting in higher overall anomaly detection performance. Practically, adopting cGGReaT could significantly strengthen cybersecurity practices by enabling accurate, interpretable, and robust anomaly detection, ultimately aiding in detecting sophisticated Cyber threats more reliably.

5.3 Limitations

Despite the promising outcomes demonstrated by our study, several limitations must be acknowledged. Firstly, the evaluation of transformer-based synthetic data generation approaches such as GReaT and cGGReaT, while thorough, revealed certain constraints in generalizability. Specifically, the effectiveness and performance of cGGReaT may vary across datasets exhibiting different cybersecurity threats or characteristics. Therefore, future studies should evaluate cGGReaT against a broader spectrum of cybersecurity datasets, encompassing more varied threat models and complexities to further validate its robustness and generalizability.

One challenge lies in the integration and deployment of transformer-based models, such as cGGReaT, into actual cybersecurity operations, including within Security Operations Centers (SOCs). Although experimental results are encouraging, evaluating their real-time application and effectiveness is essential. Future studies should investigate real-time usage, emphasizing computational demands, latency issues, and integration difficulty.

Finally, the substantial computational requirements and intricacy associated with training transformer models (LLMs) present considerable challenges, particularly in scaling for extensive datasets in real-time cybersecurity scenarios. Future investigations could prioritize enhancing these models by developing more efficient architectures or adopting incremental training techniques, thereby potentially boosting computational effectiveness and easing their implementation in environments with limited resources. Although our concept drift approach effectively adapts to dynamic changes in data distributions, it remains computationally complex, limiting its scalability and efficiency. Future work will focus on optimizing the method to reduce computational overhead while preserving its adaptability.

6 Conclusion

This paper tackles challenges in anomaly detection by addressing dataset imbalances, concept drift handling and enhancing the interpretability of AI-based cybersecurity decisions. Our approach, cGGReaT, combines generative AI methods cGAN and GReaT for effective data balancing, improving performance across classification algorithms such as XGBoost, CatBoost, LightGBM, and Extra Trees. By using explainable AI techniques SHAP and Local LIME, we increase transparency in model predictions, highlighting significant feature contributions. Our experiments on the UNSW-NB15 and CIC-IDS2017 datasets show that cGGReaT outperforms conventional resampling techniques. These results validate cGGReaT as an effective method for enhancing intrusion detection systems by improving model robustness and interpretability. Our technique currently focuses on the cyber-security domain, and its extension to new domains requires retraining the model. In future work, we will apply the developed cGGReaT to real-world business analytics in the SME (Small and Medium-Sized Enterprise) domain. This is particularly relevant as SMEs face significant data imbalance issues in detecting challenges such as cyberattacks and financial anomalies.

Acknowledgements The authors would like to express their sincere gratitude to Helge Janicke for his valuable support in this project.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Data Availability The datasets employed in this article are publicly available online at https://www.unb.ca/cic/datasets/ids-2017.html and https://research.unsw.edu.au/projects/unsw-nb15-dataset.

Declarations

Conflict of interest Authors declare no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Sarker, I.H.: AI-Driven Cybersecurity and Threat Intelligence. Springer, Berlin (2024)
2. Wang, S., Yao, X.: Multiclass imbalance problems: Analysis and potential solutions. IEEE Trans. Syst. Man Cybern. B Cybern. **42**(4), 1119–1130 (2012)
3. Tao, X., Li, Q., Guo, W., Ren, C., He, Q., Liu, R., Zou, J.: Adaptive weighted over-sampling for imbalanced datasets based on density peaks clustering with heuristic filtering. Inf. Sci. **519**, 43–73 (2020)
4. Chawla, N.V., Japkowicz, N., Kotcz, A.: Special issue on learning from imbalanced data sets. ACM SIGKDD Explor. Newsl. **6**(1), 1–6 (2004)

5. Fernández, A., López, V., Galar, M., Del Jesus, M.J., Herrera, F.: Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches. Knowl.-Based Syst. **42**, 97–110 (2013)
6. Castro, C.L., Braga, A.P.: Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. IEEE Trans. Neural Netw. Learn. Syst. **24**(6), 888–899 (2013)
7. Ghazikhani, A., Monsefi, R., Sadoghi Yazdi, H.: Online neural network model for non-stationary and imbalanced data stream classification. Int. J. Mach. Learn. Cybern. **5**, 51–62 (2014)
8. Kotsiantis, S., Kanellopoulos, D., Pintelas, P., et al.: Handling imbalanced datasets: A review. GESTS Int. Trans. Comput. Sci. Eng. **30**(1), 25–36 (2006)
9. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
10. Han, H., Wang, W.-Y., Mao, B.-H.: Borderline-smote: a new over-sampling method in imbalanced data sets learning. In: International Conference on Intelligent Computing, pp. 878–887 (2005). Springer
11. He, H., Bai, Y., Garcia, E.A., Li, S.: Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 1322–1328 (2008). Ieee
12. Ganganwar, V.: An overview of classification algorithms for imbalanced datasets. Int. J. Emerging Technol. Adv. Eng. **2**(4), 42–47 (2012)
13. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
14. Ahsan, R., Shi, W., Ma, X., Lee Croft, W.: A comparative analysis of cgan-based oversampling for anomaly detection. IET Cyber-Phys. Syst.: Theory Appl. **7**(1), 40–50 (2022)
15. Li, W., Xu, L., Liang, Z., Wang, S., Cao, J., Lam, T.C., Cui, X.: Jdgan: Enhancing generator on extremely limited data via joint distribution. Neurocomputing **431**, 148–162 (2021)
16. Dlamini, G., Fahim, M.: Dgm: a data generative model to improve minority class presence in anomaly detection domain. Neural Comput. Appl. **33**, 13635–13646 (2021)
17. Borisov, V., Seßler, K., Leemann, T., Pawelczyk, M., Kasneci, G.: Language models are realistic tabular data generators. arXiv preprint arXiv:2210.06280 (2022)
18. Corchado, J.M., López, S., Garcia, R., Chamoso, P., et al.: Generative artificial intelligence: fundamentals. Adv. Distrib. Comput. Artif. Intell. J. **12**(1), 31704 (2023)
19. Solaiman, I., Brundage, M., Clark, J., Askell, A., Herbert-Voss, A., Wu, J., Radford, A., Krueger, G., Kim, J.W., Kreps, S., et al.: Release strategies and the social impacts of language models. arXiv preprint arXiv:1908.09203 (2019)
20. Sheikholeslami, S., Ghasemirahni, H., Payberah, A.H., Wang, T., Dowling, J., Vlassov, V.: Utilizing large language models for ablation studies in machine learning and deep learning. In: Proceedings of the 5th Workshop on Machine Learning and Systems, pp. 230–237 (2025)
21. Bhuiyan, M.H., Alam, K., Shahin, K.I., Farid, D.M.: A deep learning approach for network intrusion classification. In: 2024 IEEE Region 10 Symposium (TENSYMP), pp. 1–6 (2024). IEEE
22. Benni, R., Totad, S., Mulimani, D., KG, K.: Impact analysis of real and virtual concept drifts on the predictive performance of classifiers. Procedia Comput. Sci. **235**, 617–627 (2024)
23. Nayak, P.A., Sriganesh, P., Rakshitha, K., Kumar, M.M., BS, P., HR, S.: Concept drift and model decay detection using machine learning algorithm. In: 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1–8 (2021). IEEE
24. Yang, L., Manias, D.M., Shami, A.: Pwpae: An ensemble framework for concept drift adaptation in iot data streams. In: 2021 Ieee Global Communications Conference (globecom), pp. 01–06 (2021). IEEE
25. Horchulhack, P., Viegas, E.K., Lopez, M.A.: A stream learning intrusion detection system for concept drifting network traffic. In: 2022 6th Cyber Security in Networking Conference (CSNet), pp. 1–7 (2022). IEEE
26. Hasan, M., Rahman, M.S., Janicke, H., Sarker, I.H.: Detecting anomalies in blockchain transactions using machine learning classifiers and explainability analysis. arXiv preprint arXiv:2401.03530 (2024)
27. Sarker, I.H., Janicke, H., Mohsin, A., Gill, A., Maglaras, L.: Explainable ai for cybersecurity automation, intelligence and trustworthiness in digital twin: Methods, taxonomy, challenges and prospects. ICT Express (2024). https://doi.org/10.1016/j.icte.2024.05.007
28. Ahmed, H.A., Hameed, A., Bawany, N.Z.: Network intrusion detection using oversampling technique and machine learning algorithms. PeerJ Comput. Sci. **8**, 820 (2022)
29. Batista, G.E., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor. Newsl **6**(1), 20–29 (2004)

30. Wang, M., Zheng, K., Yang, Y., Wang, X.: An explainable machine learning framework for intrusion detection systems. IEEE Access **8**, 73127–73141 (2020)
31. Gu, J., Lu, S.: An effective intrusion detection approach using svm with naïve bayes feature embedding. Comput. Secur. **103**, 102158 (2021)
32. Elmasri, T., Samir, N., Mashaly, M., Atef, Y.: Evaluation of cicids2017 with qualitative comparison of machine learning algorithm. In: 2020 IEEE Cloud Summit, pp. 46–51 (2020). IEEE
33. Kasongo, S.M., Sun, Y.: Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset. J. Big Data **7**(1), 105 (2020)
34. Bagui, S., Li, K.: Resampling imbalanced data for network intrusion detection datasets. J. Big Data **8**(1), 6 (2021)
35. Fu, Y., Du, Y., Cao, Z., Li, Q., Xiang, W.: A deep learning model for network intrusion detection with imbalanced data. Electronics **11**(6), 898 (2022)
36. Rao, Y.N., Suresh Babu, K.: An imbalanced generative adversarial network-based approach for network intrusion detection in an imbalanced dataset. Sensors **23**(1), 550 (2023)
37. Yang, H., Xu, J., Xiao, Y., Hu, L.: Spe-acgan: A resampling approach for class imbalance problem in network intrusion detection systems. Electronics **12**(15), 3323 (2023)
38. Somepalli, G., Goldblum, M., Schwarzschild, A., Bruss, C.B., Goldstein, T.: Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. arXiv preprint arXiv:2106.01342 (2021)
39. Kossen, J., Band, N., Lyle, C., Gomez, A.N., Rainforth, T., Gal, Y.: Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. Adv. Neural. Inf. Process. Syst. **34**, 28742–28756 (2021)
40. Yin, P., Neubig, G., Yih, W.-t., Riedel, S.: Tabert: Pretraining for joint understanding of textual and tabular data. arXiv preprint arXiv:2005.08314 (2020)
41. Padhi, I., Schiff, Y., Melnyk, I., Rigotti, M., Mroueh, Y., Dognin, P., Ross, J., Nair, R., Altman, E.: Tabular transformers for modeling multivariate time series. In: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3565–3569 (2021). IEEE
42. Yan, M.M.W.: Accurate detecting concept drift in evolving data streams. ICT Express **6**(4), 332–338 (2020)
43. Andresini, G., Pendlebury, F., Pierazzi, F., Loglisci, C., Appice, A., Cavallaro, L.: Insomnia: Towards concept-drift robustness in network intrusion detection. In: Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security, pp. 111–122 (2021)
44. Pesaranghader, A., Viktor, H.L.: Fast hoeffding drift detection method for evolving data streams. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 96–111 (2016). Springer
45. Qiao, H., Novikov, B., Blech, J.O.: Concept drift analysis by dynamic residual projection for effectively detecting botnet cyber-attacks in iot scenarios. IEEE Trans. Industr. Inf. **18**(6), 3692–3701 (2022). https://doi.org/10.1109/TII.2021.3108464
46. Zubair, M., Janicke, H., Mohsin, A., Maglaras, L., Sarker, I.H.: Automated sensor node malicious activity detection with explainability analysis. Sensors **24**(12), 3712 (2024). https://doi.org/10.3390/s24123712
47. Le, T.-T.-H., Prihatno, A.T., Oktian, Y.E., Kang, H., Kim, H.: Exploring local explanation of practical industrial ai applications: A systematic literature review. Appl. Sci. **13**(9), 5809 (2023)
48. Patil, S., Varadarajan, V., Mazhar, S.M., Sahibzada, A., Ahmed, N., Sinha, O., Kumar, S., Shaw, K., Kotecha, K.: Explainable artificial intelligence for intrusion detection system. Electronics **11**(19), 3079 (2022)
49. Barnard, P., Marchetti, N., DaSilva, L.A.: Robust network intrusion detection through explainable artificial intelligence (xai). IEEE Netw. Lett. **4**(3), 167–171 (2022)
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł, Polosukhin, I.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30**, (2017)
51. Moustafa, N., Slay, J.: Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS), pp. 1–6 (2015). IEEE
52. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., et al.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
53. Doquire, G., Verleysen, M.: Mutual information-based feature selection for multilabel classification. Neurocomputing **122**, 148–155 (2013)

54. Information Theory and Statistics, pp. 347–408. John Wiley & Sons, Ltd (2005). Chap. 11. https://doi.org/10.1002/047174882X.ch11 . https://onlinelibrary.wiley.com/doi/abs/10.1002/047174882X.ch11
55. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Routledge, ??? (2018)
56. Abdelkhalek, A., Mashaly, M.: Addressing the class imbalance problem in network intrusion detection systems using data resampling and deep learning. J. Supercomput. **79**(10), 10611–10644 (2023)
57. Mohammed, R., Rawashdeh, J., Abdullah, M.: Machine learning with oversampling and undersampling techniques: overview study and experimental results. In: 2020 11th International Conference on Information and Communication Systems (ICICS), pp. 243–248 (2020). IEEE
58. Drummond, C., Holte, R.C., et al.: C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on Learning from Imbalanced Datasets II, vol. 11 (2003)
59. Yi, X., Xu, Y., Hu, Q., Krishnamoorthy, S., Li, W., Tang, Z.: Asn-smote: a synthetic minority oversampling method with adaptive qualified synthesizer selection. Complex Intell. Syst. **8**(3), 2247–2272 (2022)
60. Blagus, R., Lusa, L.: Smote for high-dimensional class-imbalanced data. BMC Bioinformatics **14**, 1–16 (2013)
61. Sanh, V.: Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
62. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
63. Sennrich, R.: Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909 (2015)
64. Holtzman, A., Buys, J., Du, L., Forbes, M., Choi, Y.: The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751 (2019)
65. Keskar, N.S., McCann, B., Varshney, L.R., Xiong, C., Socher, R.: Ctrl: A conditional transformer language model for controllable generation. arXiv preprint arXiv:1909.05858 (2019)
66. Chakravarti, I.M., Laha, R.G., Roy, J.: Handbook of methods of applied statistics. Wiley Series in Probability and Mathematical Statistics (USA) eng (1967)
67. Scikit-learn: sklearn.ensemble.ExtraTreesClassifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html. Accessed: 2024-08-16
68. Ibrahim, A.A., Ridwan, R.L., Muhammed, M.M., Abdulaziz, R.O., Saheed, G.A.: Comparison of the catboost classifier with other machine learning methods. Int. J. Adv. Comput. Sci. Appl. **11**(11), 738–748 (2020)
69. Jin, D., Lu, Y., Qin, J., Cheng, Z., Mao, Z.: Swiftids: Real-time intrusion detection system based on lightgbm and parallel intrusion detection mechanism. Comput. Secur. **97**, 101984 (2020)
70. Aziz, M.N., Ahmad, T.: Clustering under-sampling data for improving the performance of intrusion detection system. JESTEC **16**, 1342–1355 (2021)
71. Widodo, A.O., Setiawan, B., Indraswari, R.: Machine learning-based intrusion detection on multi-class imbalanced dataset using smote. Procedia Comput. Sci. **234**, 578–583 (2024)
72. Hariharan, S., Rejimol Robinson, R., Prasad, R.R., Thomas, C., Balakrishnan, N.: Xai for intrusion detection system: comparing explanations based on global and local scope. J. Comput. Virol. Hack. Tech. **19**(2), 217–239 (2023)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.