MULTISTAGE CLASSIFICATION IN SDN SECURITY: A MACHINE LEARNING APPROACH USING REAL-WORLD DATA FOR ENHANCED INTRUSION AND VULNERABILITY DETECTION

Ndabuye Sengayo[1], Thomas Basikolo[2]
[1]Independent ML Researcher, Tanzania, [2]Telecommunication Standardization Bureau (TSB), International Telecommunication Union, Switzerland,

NOTE: Corresponding author: Ndabuye Sengayo (sndabuye@gmail.com)

Abstract *– The evolution from conventional network frameworks to Software-Defined Networks (SDNs) marks a pivotal shift in network management. SDNs offer a unique architecture where a centralized controller orchestrates network traffic, separating the control plane from the data plane. This architecture not only brings enhanced flexibility, agility, and scalability to network operations but also introduces specific security challenges, primarily due to the centralization of control, which can be a potential Single Point of Failures (SPOFs). Addressing these security concerns, this paper introduces an innovative machine learning-based Intrusion Detection System (IDS) tailored for SDNs, by proposing a three-stage CatBoost Classifier (3CC). This classifier is specifically designed for efficient and effective intrusion and vulnerability detection within SDN environments. The multistage aspect of the classifier allows for refined and detailed analysis, catering to a diverse array of attack vectors. Another key feature of our approach is the utilization of real-world data from SDN environments, which ensures that our model is tested and validated under realistic and current conditions. Furthermore, the integration of GPU acceleration significantly enhances the training speed of our model, a critical factor given the voluminous nature of network data. The proposed model, 3CC, demonstrates remarkable performance in our evaluations, achieving an accuracy of 99.8966%. This high level of precision in detecting and identifying various network intrusions and vulnerabilities underlines the efficacy of our approach. Our paper contributes significantly to the field of SDN security, offering a sophisticated, multistage classification solution that leverages the latest in machine learning techniques and real-world applicability. The results of this study not only underscore the potential of machine learning in network security but also pave the way for more advanced, efficient, and reliable security mechanisms in SDN environments.*

Keywords *– CatBoost, GPU acceleration, Intrusion Detection System (IDS), machine learning, Software Defined Networks (SDNs), stratified k-fold.*

1. INTRODUCTION

In the modern era of digital transformation, transition from traditional network frameworks to SDNs represents a significant paradigm shift in network management architecture [1-3]. SDNs, characterized by their centralized control mechanism, have redefined the landscape of network traffic management [4]. This modern architecture, which strategically decouples the control plane from the data plane, has not only enhanced network flexibility, agility, and scalability [5] but has also introduced unique security challenges, particularly due to the centralization of control functions, potentially creating single points of failure (SPOFs) [6].

At the heart of SDN architecture lies its three critical components: the application layer, housing software applications dictating network behavior; the control layer, featuring a centralized controller translating application requirements into network instructions; and the infrastructure layer or the data plane, consisting of physical and virtual network devices that facilitate data packet forwarding [7]. While this modular setup and programmability of the controller have revolutionized network management, it simultaneously exposes the network to new vulnerabilities [8].

A primary security concern in SDNs is the centralized control plane [9]. This concentration of control functions in a single controller presents a significant risk, as unauthorized access or attacks can lead to severe consequences, such as manipulation of network policies, traffic redirection, or communication disruptions [6]. Thus, securing the controller is pivotal for the overall security of the SDN [10]. SDN vulnerabilities are mostly

examined in five major axes of the SDN according to [11], the application layer, control layer, APIs, protocols and infrastructure layer. The biggest concern in SDN vulnerabilities is the Distributed Denial of Service (DDoS) attack [12]. This kind of attack involves overwhelming the SDN infrastructure with a high volume of traffic from various sources [13]; it can disrupt normal network operations by targeting the SDN controller or the underlying network elements.

To counter such threats, various Intrusion Detection Systems (IDSs) have been proposed, typically categorized into anomaly-based systems, which detect deviations from normal operations, and signature-based systems, which identify known attack patterns [14]. While multiple software and hardware techniques have been explored for attack detection, machine learning and deep learning techniques have emerged as particularly efficient and cost-effective solutions [15 -19]. However, most ML-focused research has predominantly concentrated on DoS attacks, leaving a gap in the coverage for other emerging attack vectors [16].

Addressing this gap, our study proposes a novel ML-based approach for a broader spectrum of attacks on SDNs. We introduce an IDS utilizing CatBoost, a renowned gradient boosting algorithm known for its strong predictive capabilities by combining simpler ML models and iteratively correcting errors [17]. Gradient boosting models have shown to be more accurate and performant than traditional ML models [20-22] in various problems as they have the ability to capture complex patterns. Thus, we present a novel GPU-accelerated three-stage CatBoost classifier (3CC) approach for enhanced intrusion and vulnerability detection in SDNs. We then benchmark the performance of our technique with other gradient boosting algorithms as they both can handle complex patterns well.

This approach, involving the combination of three CatBoost models trained on a GPU with different subsets of data representing similar attacks, has shown promising results, outperforming conventional methods.

Our contribution through this paper is threefold:

a) we introduce a **multistage classification system** to effectively account for different types of attacks;
b) we utilize newly acquired **real-world data from SDN environments**, thus presenting a more realistic and practical ML solution;
c) we employ **GPU acceleration** to significantly boost the training speed of our proposed ML model, an essential feature given the typically large volumes of networking data [18].

This paper is structured as follows: Section 2 discusses related work in the field, Section 3 details the methodology of our approach, Section 4 presents experimental results and discussions, and finally, Section 5 concludes the paper.

2. RELATED WORK

The advent of Software Defined Networks (SDNs) has revolutionized network management, prompting significant research into intrusion and vulnerability detection. In the realm of machine learning and deep learning-based IDS for SDNs, several noteworthy contributions have been made, each addressing various aspects of network security.

In the context of DDoS attack detection, a study by [19] utilized Support Vector Machines (SVMs) to analyze packet-in messages, employing entropy measures to extract key features. Their model, tested with DARPA1999 simulated DDoS attacks, achieved notable accuracy and efficiency. Similarly, [14] developed a hybrid ML technique for a flow-based IDS, integrating C4.5 algorithm (which is used to create decision trees) and K-Means, yielding an impressive 97.66% accuracy in classifying normal and attack flows.

Exploring botnet detection from flow data, [23] proposed a two-module machine learning approach. The first module extracts flow traces, while the second retrieves historical flows, demonstrating a 97% success rate for known botnets and 90% for unknown ones using decision trees and publicly available botnet data. Addressing IDS limitations, [24] proposed an IDS employing LightGBM and an autoencoder (LightGBM-AE) for feature selection and detection, achieving an 89.82% accuracy with the NSL-KDD dataset.

The use of SVM was also explored in [25], where selective logging for IP traceback was combined with the SVM. Employing the full NSL-KDD dataset, the study achieved accuracies of 95.98% and 87.74%. A deep learning-based multi-vector DDoS detection system was proposed in [26], implemented atop an SDN controller, which showed high accuracy with a low false-positive rate.

The concept of Hybrid Feature Selection using LightGBM (HSF-LGBM) was introduced in [27], involving a two-phase feature reduction approach to optimize feature subset selection. This model achieved an accuracy and F1 score of 98.72% and 98.23%, respectively. [28] presented a deep learning-based IDS, combining CNN and BiLSTM with an attention mechanism, demonstrating exceptional accuracy with the CICIDS2018 and Edge IIoT datasets.

In [29], a new approach utilized Deep Convolutional Generative Adversarial Networks (DCGAN) for data augmentation on InSDN and Edge IIoT datasets, followed by a feature extraction process using a CenterNet-based approach and a ResNet152V2 with Slime Mold Algorithm (SMA)-based deep learning model for categorizing assaults. They attained an overall accuracy of 99.65% and f1-score of 99.68%. A drawback to this study is limitation of realistic attack samples. For instance, they augmented all attack samples to match an equal number, 68 424. Some attacks like web-attack and User to Root (U2R) had 192 and 17 samples respectively, this kind of solution cannot be reliable in real environments due to the reliance on unreal data, both simulated and augmented.

Addressing the need for an IDS for attack and anomaly identification in the IoT systems, authors in [30] proposed a deep learning-based method called Deep Belief Network (DBN) algorithm using the CICIDS 2017 dataset for the performance analysis of the present IDS model. The proposed method produced excellent results in in relation to accuracy, recall, precision, f1-score, and detection rate. It achieved 99.37% accuracy for normal class, 97.93% for botnet class, 97.71% for brute force class, 96.67% for DoS/DDoS class, 96.37% for infiltration class, 97.71% for ports can class and 98.37% for Web attack. Although this method achieved very good performance, the limitation is that attacks in the dataset used were from simulations [31].

While there has been significant research in ML/DL-based IDS for SDNs, as surveyed in [32], a common limitation is the reliance on similar datasets, primarily NSL-KDD, or simulated, non-realistic SDN environments [33], and the number of attacks within the datasets. This highlights a gap in the development of IDS using real user data encompassing a diverse range of attacks.

Our study aims to fill this gap by proposing a unique technique (3CC) for IDS, leveraging real user data to enhance performance against a wide set of attacks and the capability to even classify attacks with very minor samples in the dataset (implying rare occurrences) like 'Heartbleed' as will be seen later in the paper. The proposed model is capable of identifying multiple classes (15). This not only complements existing solutions but also provides a more realistic and comprehensive approach to SDN security.

3. METHOD

This section describes the proposed method for intrusion detection in SDNs. In summary, we firstly establish experimental clusters of the labels in the dataset along with feature importance using a single CatBoost model. Then, data is grouped based on the label clusters derived from their similarities. Afterwards, different CatBoost models are trained for different subsets of the data and finally, the wrapped algorithm is evaluated on unseen data.

Figure 1 below shows a high-level representation of the approach. It encompasses the following major stages:

1) Data preprocessing
2) Establishing labels clusters and feature importance
3) Training three CatBoost models for each of the three data subsets.
4) Wrapping and evaluation.

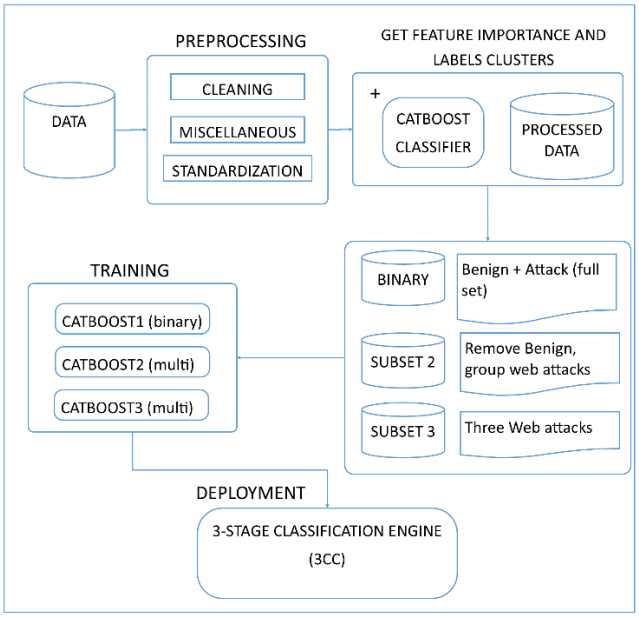

Figure 1 - Proposed CatBoost-based IDS approach

3.1 Data description

The data used in this study was provided by ULAK Communications [34] in the context of ITU AI/ML in 5G Challenge 2023 [35] titled "*Intrusion and Vulnerability Detection in Software-Defined Networks (SDN)*". The dataset combines a portion of real users obtained from ULAK Communications' SD-WAN environment combined with the CICIDS 2027 dataset [31]. The dataset encompasses four primary types of samples: *Normal flow data*, *DDoS flow data, Malware flow data,* and *web-based flow data.* Overall, the dataset contains over 2.2 million samples with 78 features before preprocessing, in which 20% of the data was used for testing.

Table 1 give a more detailed explanation on the types of attacks that are used in this study. In Figure 2, we show the distribution of the labels within the dataset used.

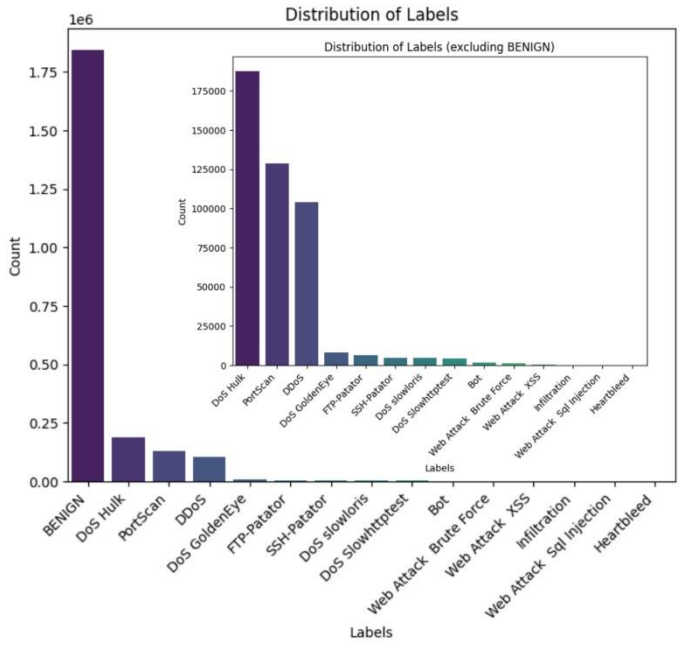

Figure 2 - Distribution of classes in the dataset (included is distribution without BENIGN for better visualization)

3.2 Data preprocessing

Data can be inconsistent, loud, excessive, incomplete, or generated from a variety of sources. Therefore, it is essential to turn raw data into knowledge that may be used for research and disclosure. In this work, we created a pipeline for preprocessing the data from its original form to a train-ready state. The subsections below elaborate important stages in the pipeline.

3.2.1 Cleansing

This is the process of detecting incorrect and/or noisy samples from the data and either correcting them using various techniques or removing them as shown in [36]. Data cleansing involves removal of the following kinds of samples:

- Duplicate samples
- Columns with a single unique value
- Columns with over 20% missing data
- Filling missing values, with their modes.

3.2.2 Miscellaneous

In this study, several techniques were applied to the data based on its inherent characteristics, as outlined below.

- Given that the data inherently should not contain negative values, any negative values (implying faulty data recording) encountered were substituted with null values.
- For instances of missing data, we replaced these values with the most frequent values (modes) in their respective categories.

3.2.3 Standardization

Standardization is a crucial preprocessing step, particularly when dealing with datasets where the data samples are spread across various scales. This was notably applicable to our dataset. The disparate range of scales can lead to biased or skewed results in machine learning algorithms, as they often assume data to be on a similar scale.

To ensure consistent feature representation and to facilitate model convergence, we employed standard scaler from sci-kit learn on all numeric features with more than two unique values. In this case, standardization was completed to the whole dataset before any modeling processes.

This technique transforms each feature to have a zero mean (μ) and unit standard deviation (σ).

$$z = \frac{x - \mu}{\sigma}$$

It uses the formula above, where x is the original value and z is the transformed value.

Table 1 - Meanings of types of attacks in the used data

Attack Name	Meaning
DoS Hulk	A volumetric attack that floods a target with a large number of requests, overwhelming its resources.
Benign	While not technically an attack, this term is sometimes used to describe legitimate traffic that can accidentally overload a system.
DDoS	Stands for Distributed Denial of Service, it is similar to DoS but originating from multiple sources, making it harder to defend against.
PortScan	Attempts to identify open ports on a target system, often a precursor to other attacks.
DoS GoldenEye	A volumetric attack that exploits vulnerabilities in the SYN-ACK handshake of TCP connections.
FTP-Patator	A tool used to brute force FTP passwords.
DoS slowloris	A low-rate attack that slowly exhausts a target's resources by establishing many long-lived connections.
DoS Slowhttptest	A tool used to perform slowloris attacks.
SSH-Patator	A tool used to brute force SSH passwords.
Web - XSS	XSS stands for Cross-Site Scripting, this is where malicious code is injected into a web page to be executed by users.
Web - Brute Force	Repeated attempts to guess passwords, often targeting login forms.
Web - SQL Injection	Injecting malicious SQL code into a web application to gain unauthorized access to a database.
Bot	A malicious software program that performs automated tasks, often used in DDoS attacks or spam campaigns.
Infiltration	Gaining unauthorized access to a system or network.
Heartbleed	Specific vulnerability in the OpenSSL cryptographic library that allows attackers to steal sensitive data.

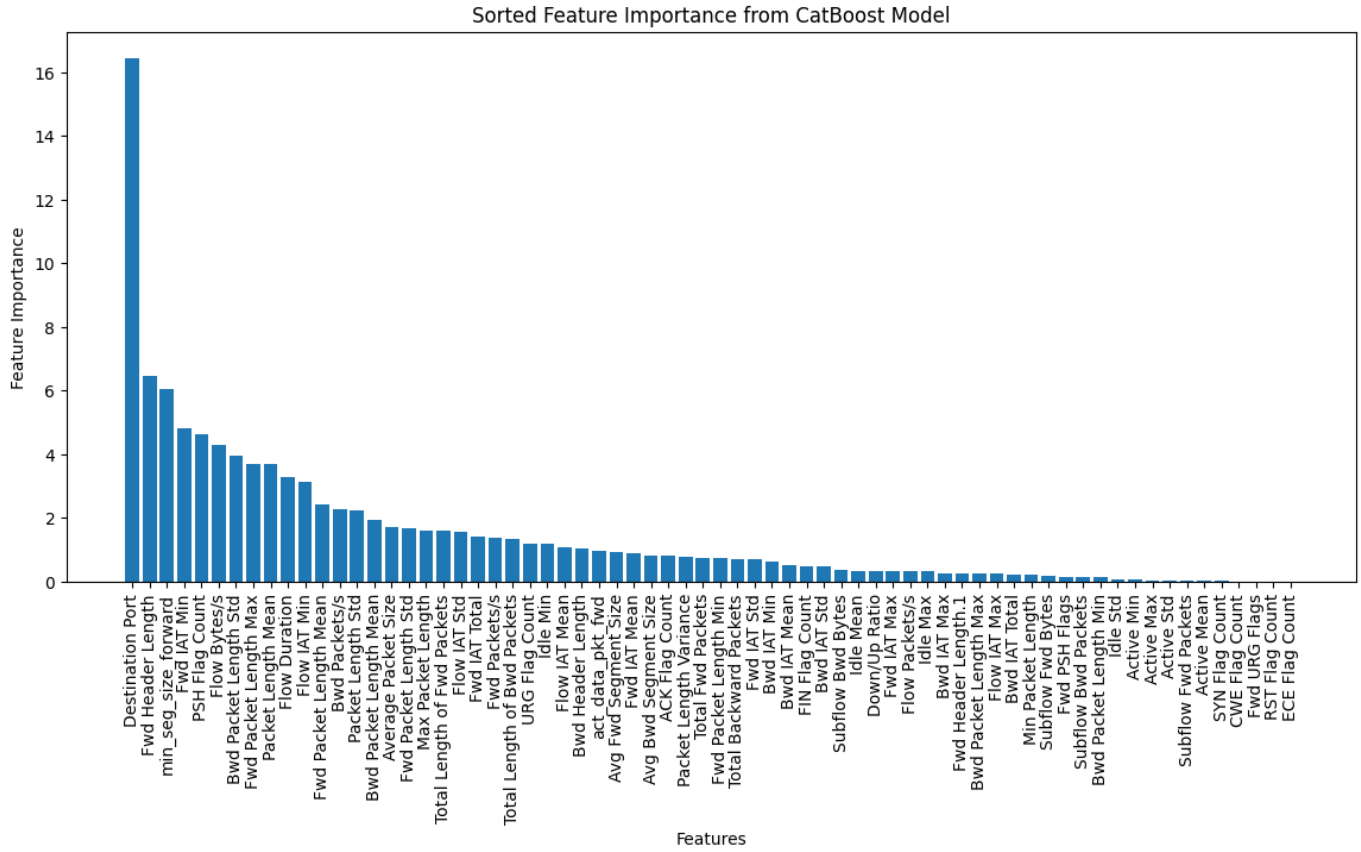

Figure 3 - Feature importance

3.3 Feature selection

Feature selection is an essential component in data preprocessing, especially in scenarios where the dataset encompasses a vast array of features. In our study, we first train a CatBoost model using the default parameters and calculate the feature importance using the Prediction Values Change (PVC). This method measures how much the prediction value changes when the values of a feature are changed. The importance of a feature is computed by averaging the changes in prediction values across all trees in the model.

This approach enabled us to quantify the significance of individual features in the prediction process. Based on the feature importance values generated by the CatBoost model, we systematically removed all features that exhibited zero importance, as they contribute minimally to the model's performance. Figure 3 illustrates a plot detailing these feature importance values as extracted by the trained CatBoost model, providing a visual representation of the contribution of each feature to the overall model efficacy. This step not only streamlines the model but also enhances its accuracy by focusing on the most relevant features.

3.4 Training procedure

This section outlines the methodologies and techniques implemented during the training phase of our study. The training procedure is a critical component in developing an effective machine learning model, and it involves several steps. We first establish clusters from available labels, train different models on created data subsets using a stratified K-fold training technique.

3.4.1 Establishing label clusters

As shown in Figure 1, we establish clusters by first training a CatBoost model on the full dataset. This gives us the likeness of the labels in the dataset, we then group these and use separate models to classify them. Table 2 shows the data subsets along with their labels used in training their respective model.

3.4.2 GPU-accelerated CatBoost classifier

CatBoost is a machine learning library for gradient boosting on decision trees. It builds decision trees and combines them to create a strong predictive model [37]. Its advantage lies in its optimizations for training speed, memory usage, and regularization techniques.

Gradient boosting models were selected due to their superior ability to capture complex patterns compared to traditional machine learning algorithms, and networking environments tend to have such kinds of data, which is why this study utilizes CatBoost.

Another very important advantage of CatBoost is its built-in support for GPU, which allows for faster training in larger datasets [38]. Therefore, this allowed us to use the T4 GPU provided in Google colab GPU runtime.

3.4.3 Stratified K-fold

Considering the unbalanced nature of the data, we opted to use stratified K-fold in training so as to maintain the proportion of the labels in each of the training phases. This is an efficient method that prevents duplication in the categories, thus reducing bias and maintaining proportion [39].

The following steps show how stratified K-fold works.

- *Initialization*
  The dataset is divided into K approximately equal-sized folds. We used k=5 since this number of folds enables good proportion of samples from minority classes in each fold.
- *Repetition*
  The entire K-fold cross-validation process is typically repeated several times and the results are averaged to obtain a more robust evaluation.
- *Stratification*
  The goal is to ensure that each fold has a similar distribution of the target classes as the entire dataset. In other words, the proportion of different classes in each fold should be representative of the overall dataset.
- *Iteration*
  The cross-validation process involves K iterations. In each iteration, one of the folds is used as the validation set, and the remaining K-1 (4 in our case) folds are used for training the model.

Therefore, each of the three CatBoost models were trained on five stratified folds of their data subsets.

Table 2 - Clusters of labels and their subsets

Labels	Set1(Full)	Set2 (452k)	Set3 (1.7k)
BENIGN	BENIGN		
DoS Hulk	Attacks	DoS Hulk	
PortScan		PortScan	
DoS slowloris		DoS slowloris	
DoS GoldenEye		DoS GoldenEye	
DoS Slowhttptest		DoS Slowhttptest	
DDoS		DDoS	
FTP-Patator		FTP-Patator	
SSH-Patator		SSH-Patator	
Bot		Bot	
Infiltration		Infiltration	
Heartbleed		Heartbleed	
Web – Sql Injection		Web Attack	Web – Sql Injection
Web – XSS			Web – XSS
Web - Brute Force			Web - Brute Force

3.4.4 Model wrapping

Our ensemble approach differs from traditional ensemble methods where multiple models are combined via voting or averaging. Instead, the final stage utilizes a sequential processing pipeline that incorporates three CatBoost classifiers. Each classifier is trained on distinct data subsets as shown in the clusters in Table 2, and a data sample can potentially be evaluated by all three models to arrive at the final label.

In essence, the first model performs a binary classification to identify benign traffic versus potential attacks. Subsequently, the second model focuses on differentiating between various attack categories within the identified attack group. Finally, the last model classifies the specific type of web attack. This approach leverages the strengths of each classifier in a sequential manner.

3.5 Evaluation techniques

Although using stratified K-fold as a training method usually gives a view of the performance of the model, it is usually better to have unseen data for testing to get the accurate performance of the model. We next discuss the techniques used to establish the performance of the developed solution.

3.5.1 Testing data

For effective evaluation, it is always safer to set aside a significant data size to be used for testing the performance of the model after training. This is the portion of the data that the model has not seen.

In this work, we set aside over 20% (over 500K samples) of the original data for testing purposes. Here, we used the 3-stage classification wrapped algorithm to make predictions on the test data.

3.5.2 Metrics

Evaluation metrics play a critical role in obtaining an optimal classifier during the training process of a classification machine learning model [40]. Table 3 describes the metrics used in this work. For the formulas, here are the long forms of the abbreviations used:

- TP = True Positive
- TN = True Negative
- FP = False Positive
- FN = False Negative

Table 3 - Evaluation metrics

Metric	Description	Formula
Accuracy	The accuracy metric gauges the proportion of accurate predictions in relation to the overall number of instances assessed.	$\frac{TP + TN}{TP + FP + TN + FN}$
Recall	Recall is employed to assess the proportion of correctly classified positive patterns among the total positive instances	$\frac{TP}{TP + TN}$
Precision	Precision is utilized to quantify the accurately predicted positive patterns in relation to the total predicted patterns within a positive class.	$\frac{TP}{TP + FP}$
F1- measure	This measurement shows the harmonic mean between the values of recall and precision. It's important and mostly when there's class imbalance in the data.	$\frac{2 * Precision * Recall}{Precision + Recall}$

4. RESULTS AND DISCUSSION

This section primarily focuses on evaluating the performance of the proposed 3CC model for intrusion and vulnerability detection in Software Defined Networks (SDNs). This section is critical as it provides empirical evidence to validate the efficacy of the 3CC approach compared to other conventional models.

4.1 Performance of 3CC

3CC demonstrated outstanding performance on unseen data, which is crucial for assessing its real-world applicability. The highlight of the results is the overall accuracy of the model, which reached 99.90% as well as the f1 score of 99.10%. This level of accuracy is remarkable as it significantly surpasses the performance of conventional models in the same domain.

Table 4 below summarizes the performance of the 3CC model in general and stage-wise.

Table 4 - Performance of 3CC and its stages

	1C	2C	3C
Accuracy	99.83%	99.94%	81.89%
F1	99.91%	99.96%	81.92%
Overall Accuracy			**99.90%**
Overall F1 Score			**99.10%**

The high accuracy indicates the model's robustness in correctly classifying various types of network intrusions and vulnerabilities, indicating its potential as a reliable tool in SDN security.

We further analyze the performance of the proposed model through the use of a confusion matrix as shown in Figure 4. Confusion matrix is crucial for evaluating the efficacy of classification models. The matrix allows us to see not only the number of correct predictions made by the model (TP and TN) but also the instances where it misclassified (FP and FN). This information is essential for understanding and improving the model's performance.

Along with the confusion matrix, Table 5 shows the classification report generated after making predictions on the unseen test data showing precision, recall and f1-score of every label. Slightly lower performances from the classification can be traced to fewer samples in web traffic in the dataset.

Table 5 - Classification report

Class	Precision	Recall	F1
BENIGN	1.00	1.00	1.00
Bot	1.00	1.00	1.00
DDoS	1.00	1.00	1.00
DoS GoldenEye	1.00	1.00	1.00
DoS Hulk	1.00	1.00	1.00
DoS Slowhttptest	0.99	0.99	0.99
DoS slowloris	0.99	1.00	0.99
FTP-Patator	1.00	1.00	1.00
Heartbleed	1.00	1.00	1.00
Infiltration	1.00	0.75	0.86
PortScan	1.00	1.00	1.00
SSH-Patator	1.00	1.00	1.00
Web Attack Brute Force	0.76	0.88	0.82
Web Attack Sql Injection	1.00	1.00	1.00
Web Attack XSS	0.56	0.34	0.43

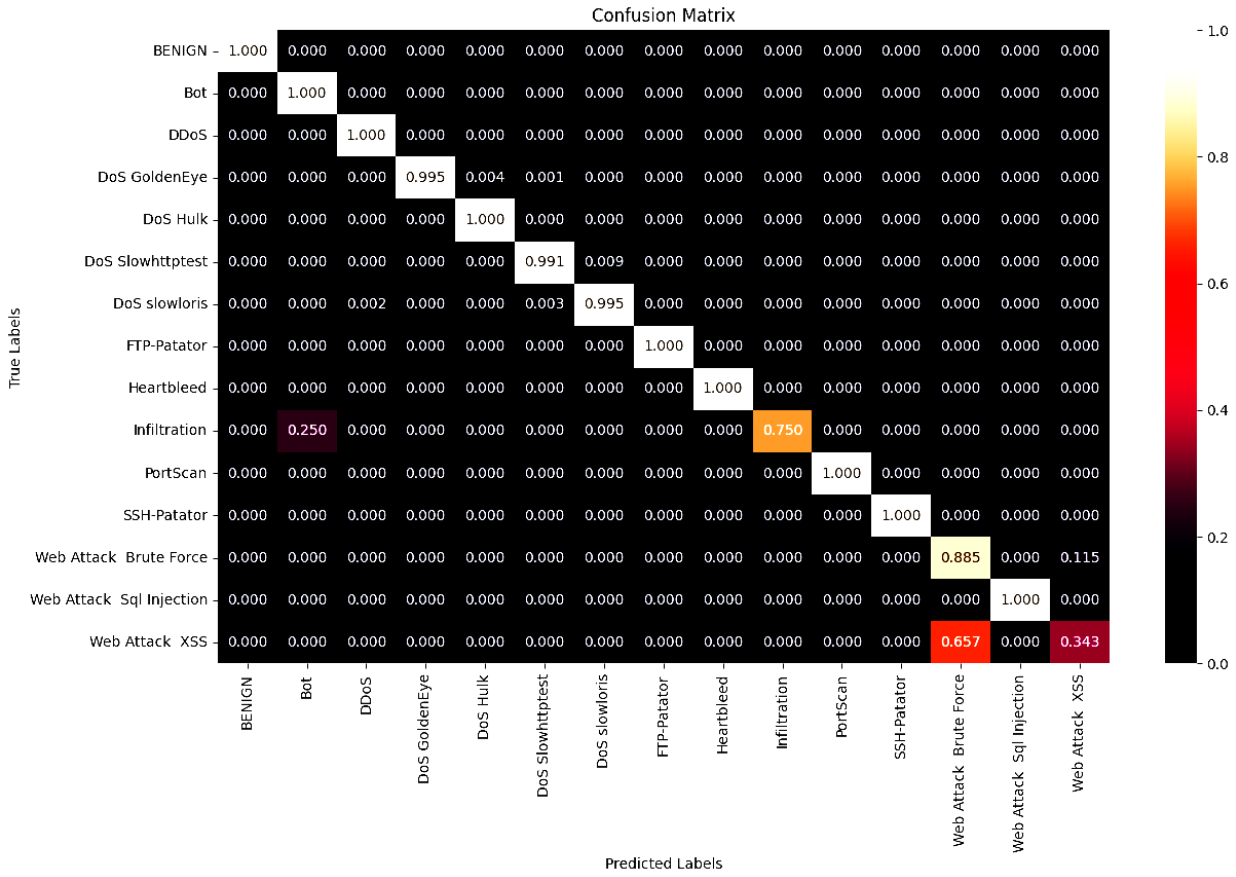

Figure 4 - Confusion matrix showing predictions from unseen data by 3CC

4.2 Comparison with other models.

To establish a comprehensive understanding of the 3CC model's performance, baseline comparisons were conducted with other gradient boosting algorithms, which are known for their effectiveness in classification tasks. We used LightGBM, HistGradient Boosting, and CatBoost. These models were trained using the same training data and evaluated using the same unseen data to ensure that the comparison is fair. We base the comparison on performance in terms of accuracy. This therefore provides a clear perspective on how the novel 3CC model stands against the other models.

Table 6 shows performance comparison between 3CC and other well-known models.

Table 6 - Comparison with other models

Model	Accuracy	F1-score
LGBM	90.69%	88.91%
HistGBoosting	99.21%	98.12%
CatBoost	99.70%	98.28%
3CC	99.89%	99.80%

From Table 6, our proposed model shows the best accuracy score. The difference in the performance of CatBoost and 3CC might seem tiny but it is significant when we dig deeper into the details as discussed below.

Figure 5 shows the confusion matrix for the single conventional CatBoost model. When compared to the confusion matrix of 3CC in Figure 4 above, we can see the superior ability of 3CC in accurately classifying different types of network flows. The matrix shows a higher number of correct predictions (both TP and TN) by the 3CC model, indicating its enhanced accuracy and reliability. The matrix for single CatBoost model in this figure reveals the tendency of CatBoost to misclassify minority flows as majority ones in some cases, leading to a higher rate of classification errors compared to 3CC. Such confusion in classifications, particularly in the context of minority flows, can significantly impact the effectiveness of an intrusion detection system. This enhanced performance of 3CC in classifying minority classes effectively translates to a more robust and reliable intrusion detection system for SDNs.

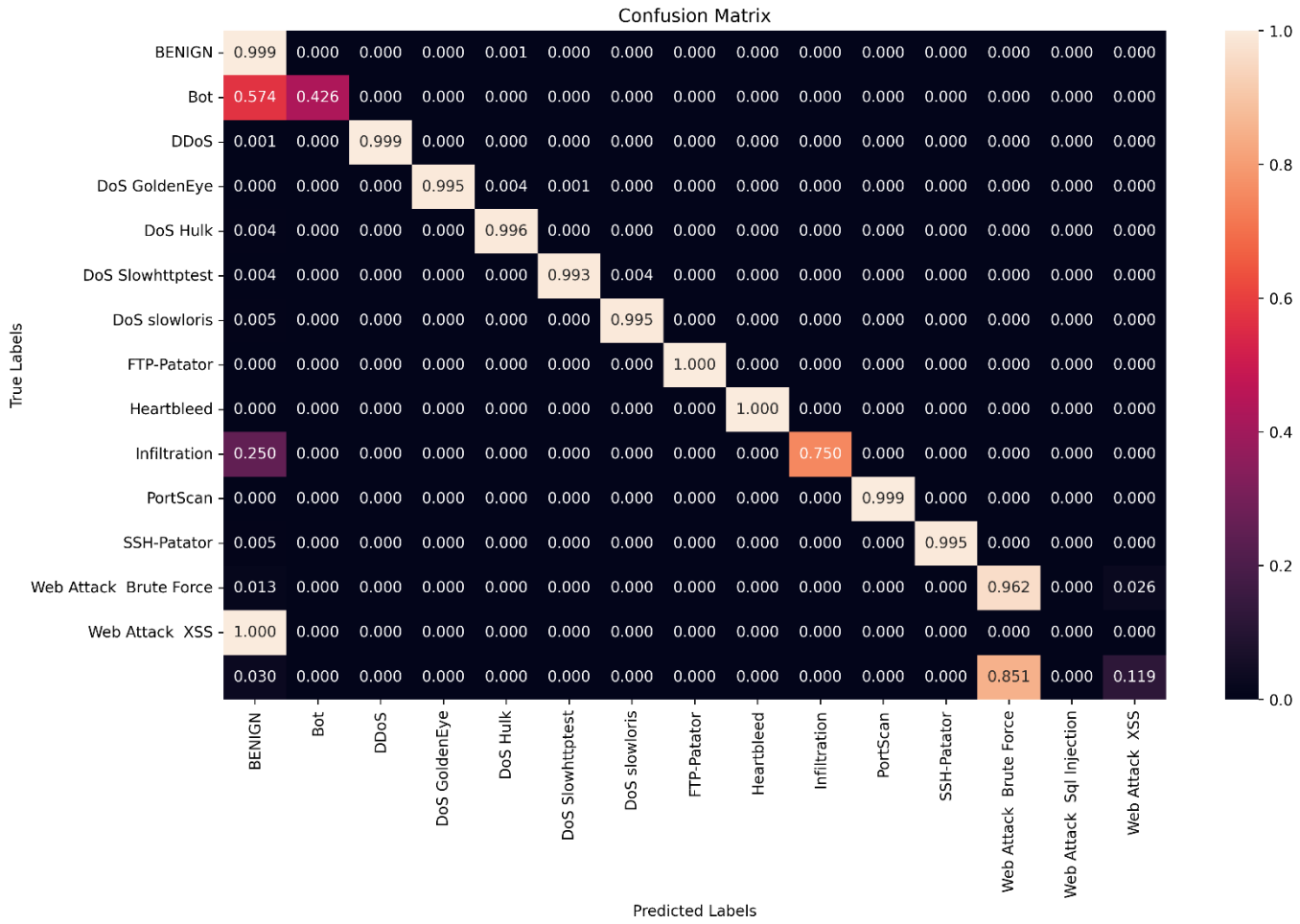

Figure 5 - Confusion matrix of a single CatBoost model

Table 7 - Comparison with other related studies

Reference	Approach	Dataset	Number. of classes	Accuracy
Sudar et al. [14]	Hybrid (C4.5 + K-Means)	NSL-KDD	6	97.66%
Parra et al. [41]	DCNN LSTM	N_BaIoT	11	94.30%
Hadem et al. [25]	SVM	NSL-KDD	6	95.98%
Raj et al. [42]	CatBoost	DDos Logs (Kaggle)	5	98.62%
Manimurugan et al. [30]	DBN	CICIDS 2017	15	99.37%
Logeswari et al. [27]	HFS-LGBM	NSL-KDD	4	98.72%
Tang et al. [24]	LightGBM-AE	NSL-KDD	2	89.82%
This paper	3CC	ULAK's Real SD-WAN data + CICIDS 2017	15	99.89%

4.3 Comparison with other studies

We have also undertaken a comparison study with other ML-based solutions in the realm of intrusion and vulnerability detection in SDNs. This comparison is vital to contextualize our study within the broader landscape of current research and to highlight the unique contributions of our solution. A significant finding from our comparison is that no other study, which utilized real data for their solutions, has achieved an accuracy level surpassing that of our proposed model. This underscores the excellent performance and novelty of our approach. The usage of real-world data in our study is crucial, as it provides a more realistic and challenging testing environment compared to simulated or synthetic datasets often used in other researches. The results in Table 7 compares the accuracy of the 3CC model with those from a variety of other studies, encompassing both machine learning and deep learning solutions. The table also highlights the techniques and data used in each of the studies.

It serves as a quick reference for readers to gauge the performance of our model relative to other high-performing solutions in the field.

5. CONCLUSION

SDNs represents a significant evolution in network management, characterized by the decoupling of the control plane from the data plane. This structural change in SDNs brings about enhanced flexibility and programmability, enabling networks to dynamically adapt to changing traffic patterns and application demands. In this paper, we introduced a novel intrusion detection approach for SDN environments: a multiclass machine learning classifier, 3CC. This model capitalizes on the inherent similarities within groups of network attacks to create a more precise and efficient detection mechanism. Some of the key features of our approach is multiclass classification in which the model is capable of classifying 15 classes with high accuracy and the use of GPU acceleration to expedite the training process, addressing the voluminous nature of networking data. Through meticulous data preprocessing and the implementation of stratified K-fold training techniques, the 3CC model achieved an outstanding accuracy of 99.8966% on unseen test data.

The classifier's performance, as detailed in the classification report, underscored its effectiveness across a variety of traffic types, confirming its robustness and reliability. Furthermore, our solution was developed using data from an actual SDN environment, which not only bolsters its empirical validity but also enhances its applicability in real-world settings.

However, it is important to acknowledge certain limitations in our study, particularly the limited data available for some attack classes and decline in performance on some classes (web attacks), which is caused by the quality and size of data in these particular classes of attacks, of which they were found to have very deep similarities, making it harder to mathematically distinguish. This limitation highlights the need for more comprehensive datasets to further refine and validate the efficacy of the 3CC model across an even broader spectrum of network attacks and scenarios.

In conclusion, the GPU-accelerated 3CC presents a novel approach in the domain of SDN security, offering a highly accurate, efficient, and practical solution for intrusion detection. Future work will focus on expanding the dataset and exploring additional features to overcome current limitations and further enhance the model's performance and applicability in diverse SDN environments.

REFERENCES

[1] A. Ben Letaifa, “SSIM and ML based QoE enhancement approach in SDN context,” *Advances in Computers*, vol. 114, pp. 151–196, Jan. 2019, doi: 10.1016/BS.ADCOM.2019.02.004.

[2] Q. Waseem, S. S. Alshamrani, K. Nisar, W. I. S. W. Din, and A. S. Alghamdi, “Future Technology: Software-Defined Network (SDN) Forensic,” *Symmetry 2021, Vol. 13, Page 767*, vol. 13, no. 5, p. 767, Apr. 2021, doi: 10.3390/SYM13050767.

[3] S. Saraswat, V. Agarwal, H. P. Gupta, R. Mishra, A. Gupta, and T. Dutta, “Challenges and solutions in Software Defined Networking: A survey,” *Journal of Network and Computer Applications*, vol. 141, pp. 23–58, Sep. 2019, doi: 10.1016/J.JNCA.2019.04.020.

[4] G. A. N. Segura, A. Chorti, and C. B. Margi, “Centralized and Distributed Intrusion Detection for Resource Constrained Wireless

SDN Networks," Mar. 2021, [Online]. Available: http://arxiv.org/abs/2103.01262

[5] "What is Software Defined Infrastructure? Definition, Types, Examples | Enterprise Tech News EM360." Accessed: Jan. 25, 2024. [Online]. Available: https://em360tech.com/tech-article/what-is-software-defined-infrastructure

[6] A. Pradhan and R. Mathew, "Solutions to Vulnerabilities and Threats in Software Defined Networking (SDN)," in *Procedia Computer Science*, Elsevier B.V., 2020, pp. 2581–2589. doi: 10.1016/j.procs.2020.04.280.

[7] N. S. Shaji and R. Muthalagu, "Survey on security aspects of distributed software-defined networking controllers in an enterprise SD-WLAN," *Digital Communications and Networks*, Sep. 2023, doi: 10.1016/J.DCAN.2023.09.004.

[8] H. Farhady, H. Lee, and A. Nakao, "Software-Defined Networking: A survey," *Computer Networks*, vol. 81, pp. 79–95, Apr. 2015, doi: 10.1016/J.COMNET.2015.02.014.

[9] M. S. Farooq, S. Riaz, and A. Alvi, "Security and Privacy Issues in Software-Defined Networking (SDN): A Systematic Literature Review," *Electronics 2023, Vol. 12, Page 3077*, vol. 12, no. 14, p. 3077, Jul. 2023, doi: 10.3390/ELECTRONICS12143077.

[10] U. Tupakula, V. Varadharajan, and P. Mishra, "Securing SDN controller and switches from attacks," *International Journal of High Performance Computing and Networking*, vol. 14, no. 1, p. 77, 2019, doi: 10.1504/IJHPCN.2019.099746.

[11] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, "Software-defined networking (SDN): a survey," *Security and Communication Networks*, vol. 9, no. 18, pp. 5803–5833, Dec. 2016, doi: 10.1002/sec.1737.

[12] M. Myint Oo, S. Kamolphiwong, T. Kamolphiwong, and S. Vasupongayya, "Advanced Support Vector Machine-(ASVM-) based detection for Distributed Denial of Service (DDoS) attack on Software Defined Networking (SDN)," *Journal of Computer Networks and Communications*, vol. 2019, 2019, doi: 10.1155/2019/8012568.

[13] K. Jia, C. Liu, Q. Liu, J. Wang, J. Liu, and F. Liu, "A lightweight DDoS detection scheme under SDN context," *Cybersecurity*, vol. 5, no. 1, p. 27, Dec. 2022, doi: 10.1186/s42400-022-00128-7.

[14] K. M. Sudar and P. Deepalakshmi, "Flow Based Intrusion Detection System for Software Defined Networking using Hybrid Machine Learning Technique," *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, no. 2S2, pp. 1026–1033, Dec. 2019, doi: 10.35940/ijitee.b1108.1292s219.

[15] N. Rawindaran, A. Jayal, E. Prakash, and C. Hewage, "Cost Benefits of Using Machine Learning Features in NIDS for Cyber Security in UK Small Medium Enterprises (SME)," *Future Internet 2021, Vol. 13, Page 186*, vol. 13, no. 8, p. 186, Jul. 2021, doi: 10.3390/FI13080186.

[16] A. Mahajan and A. Bhandari, "Attacks in Software-Defined Networking: A Review," *SSRN Electronic Journal*, Mar. 2020, doi: 10.2139/SSRN.3564048.

[17] A. V. Konstantinov and L. V. Utkin, "Interpretable machine learning with an ensemble of gradient boosting machines," *Knowl Based Syst*, vol. 222, p. 106993, Jun. 2021, doi: 10.1016/J.KNOSYS.2021.106993.

[18] D. Zhou, Z. Yan, Y. Fu, and Z. Yao, "A survey on network data collection," *Journal of Network and Computer Applications*, vol. 116, pp. 9–23, Aug. 2018, doi: 10.1016/J.JNCA.2018.05.004.

[19] D. Li, C. Yu, Q. Zhou, and J. Yu, "Using SVM to Detect DDoS Attack in SDN Network," *IOP Conf Ser Mater Sci Eng*, vol. 466, no. 1, Dec. 2018, doi: 10.1088/1757-899X/466/1/012003.

[20] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of gradient boosting algorithms," *Artif Intell Rev*, vol. 54, no. 3, pp. 1937–1967, Mar. 2021, doi: 10.1007/S10462-020-09896-5/METRICS.

[21] "Gradient Boosting Trees vs. Random Forests | Baeldung on Computer Science." Accessed: Jan. 26, 2024. [Online]. Available: https://www.baeldung.com/cs/gradient-boosting-trees-vs-random-forests

[22] N. Aziz, E. A. P. Akhir, I. A. Aziz, J. Jaafar, M. H. Hasan, and A. N. C. Abas, "A Study on Gradient

Boosting Algorithms for Development of AI Monitoring and Prediction Systems," *2020 International Conference on Computational Intelligence, ICCI 2020*, pp. 11–16, Oct. 2020, doi: 10.1109/ICCI51257.2020.9247843.

[23] F. Tariq and S. Baig, "Machine Learning Based Botnet Detection in Software Defined Networks," *International Journal of Security and Its Applications*, vol. 11, no. 11, pp. 1–12, Nov. 2017, doi: 10.14257/IJSIA.2017.11.11.01.

[24] C. Tang, N. Luktarhan, and Y. Zhao, "An efficient intrusion detection method based on LightGBM and autoencoder," *Symmetry (Basel)*, vol. 12, no. 9, Sep. 2020, doi: 10.3390/SYM12091458.

[25] P. Hadem, D. K. Saikia, and S. Moulik, "An SDN-based Intrusion Detection System using SVM with Selective Logging for IP Traceback," *Computer Networks*, vol. 191, May 2021, doi: 10.1016/J.COMNET.2021.108015.

[26] Q. Niyaz, W. Sun, and A. Y. Javaid, "A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN)," *ICST Transactions on Security and Safety*, vol. 4, no. 12, p. 153515, Dec. 2017, doi: 10.4108/EAI.28-12-2017.153515.

[27] G. Logeswari, S. Bose, and T. Anitha, "An Intrusion Detection System for SDN Using Machine Learning," *Intelligent Automation and Soft Computing*, vol. 35, no. 1, pp. 867–880, 2023, doi: 10.32604/iasc.2023.026769.

[28] V. Hnamte and J. Hussain, "DCNNBiLSTM: an efficient hybrid deep learning-based intrusion detection system," *Telemat. Inform. Rep.*, vol. 10, Jun. 2023, doi: 10.1016/j.teler.2023.100053.

[29] M. Maddu and Y. N. Rao, "Network intrusion detection and mitigation in SDN using deep learning models," *Int J Inf Secur*, pp. 1–14, Oct. 2023, doi: 10.1007/S10207-023-00771-2/METRICS.

[30] S. Manimurugan, S. Al-Mutairi, M. M. Aborokbah, N. Chilamkurti, S. Ganesan, and R. Patan, "Effective attack detection in internet of medical things smart environment using a deep belief neural network," *IEEE Access*, vol. 8, pp. 77396–77404, 2020, doi: 10.1109/ACCESS.2020.2986013.

[31] "IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB." Accessed: Jan. 30, 2024. [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html

[32] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer Peer Netw Appl*, vol. 12, no. 2, pp. 493–501, Mar. 2019, doi: 10.1007/S12083-017-0630-0/METRICS.

[33] M. S. Elsayed, N. A. Le-Khac, and A. D. Jurcut, "InSDN: A novel SDN intrusion dataset," *IEEE Access*, vol. 8, pp. 165263–165284, 2020, doi: 10.1109/ACCESS.2020.3022633.

[34] "ULAK HABERLEŞME A.Ş. ENG." Accessed: Jan. 25, 2024. [Online]. Available: https://ulakcommunications.com/index.php/en/

[35] "ITU-ML5G-PS-006: Intrusion and Vulnerability Detection in Software-Defined Networks (SDN)." Accessed: Jan. 25, 2024. [Online]. Available: https://challenge.aiforgood.itu.int/match/matchitem/81

[36] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proceedings*, vol. 3, no. 1, pp. 91–99, Jun. 2022, doi: 10.1016/J.GLTP.2022.04.020.

[37] A. A. Ibrahim, R. L. Ridwan, M. M. Muhammed, R. O. Abdulaziz, and G. A. Saheed, "Comparison of the CatBoost Classifier with other Machine Learning Methods," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 11, pp. 738–748, 2020, doi: 10.14569/IJACSA.2020.0111190.

[38] A. Lazar, A. Sim Kesheng Wu, L. Berkeley, A. Sim, and K. Wu, "GPU-based Classification for Wireless Intrusion Detection," *SNTA 2021 - Proceedings of the 2021 Systems and Network Telemetry and Analytics, co-located with HPDC 2021*, pp. 27–31, Jun. 2021, doi: 10.1145/3452411.3464445.

[39] M. K. Mayangsari, I. Syarif, and A. Barakbah, "Evaluation of Stratified K-fold Cross Validation for Predicting Bug Severity in Game Review Classification," *Kinetik: Game Technology, Information System, Computer*

Network, Computing, Electronics, and Control, vol. 8, no. 3, pp. 651–662, Aug. 2023, doi: 10.22219/KINETIK.V8I3.1740.

[40] H. M and S. M.N, "A Review on Evaluation Metrics for Data Classification Evaluations," *International Journal of Data Mining & Knowledge Management Process*, vol. 5, no. 2, pp. 01–11, Mar. 2015, doi: 10.5121/ijdkp.2015.5201.

[41] G. De La Torre Parra, P. Rad, K. K. R. Choo, and N. Beebe, "Detecting Internet of Things attacks using distributed deep learning," *Journal of Network and Computer Applications*, vol. 163, p. 102662, Aug. 2020, doi: 10.1016/J.JNCA.2020.102662.

[42] A. Raj, K. Saivenu, M. Irteqa Ahmed, and A. Kanavalli, "Detection and mitigation of botnet based DDoS attacks using CatBoost machine learning algorithm in SDN environment," *International Journal of Advanced Technology and Engineering Exploration*, vol. 8, no. 76, pp. 2394–7454, 2021, doi: 10.19101/IJATEE.2021.874021.

AUTHORS

Ndabuye Sengayo is an award-winning machine learning engineer. He works as a data scientist with consulting firms. He has won various national and international awards, including best technology champion youth in Tanzania 2022, and best cloud and AI solutions with Huawei Cloud (HDC 2022). He has worked as a trainer on 4IR technologies at the University of Dodoma where he received a bachelor's degree (BSc.) in software engineering in Tanzania and was named as the college best student. His fields of interest include cloud and AI and their applications in advancing scientific frontiers.

Thomas Basikolo is a programme officer with the ITU Telecommunication Standardization Bureau (TSB). He coordinates and manages the ITU AI/ML in 5G Challenge and is an advisor to the ITU-T Focus Group on Autonomous Networks. Prior to joining ITU, he worked as a research engineer in the Engineering Department of Microwave Factory Co., Ltd, Tokyo, Japan. He received a PhD in electrical and computer engineering from Yokohama National University, Japan. He is a recipient of multiple best paper awards, the IEEE AP-S Japan Student Award and the Young Engineer of the year award by IEEE AP-S Japan in 2018. He has co-authored peer-reviewed journal and conference papers, predominantly in the areas of wireless communications and antenna engineering. He serves as a reviewer of IEEE and IEICE journals. His interests include machine learning, deep learning and network science, and their applications in wireless networks.

