TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 24, No. 4, August 2026, pp. 1204~1215
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v24i4.27561

Real-time anomaly detection system using best performed machine learning model

Victor Mathebula, Bukohwo Michael Esiefarienrhe
Department of Computer Science and Information Systems, North-West University, Mahikeng, South Africa

Article Info

Article history:

Received Sep 21, 2025
Revised Apr 9, 2026
Accepted May 25, 2026

Keywords:

Anomaly detection
Artificial intelligence
Hybrid machine learning
Knowledge discoveries in databases
Machine learning algorithms

ABSTRACT

Effective anomaly detection is critical for protecting organizational networks against increasingly sophisticated cyber threats. However, most machine learning-based intrusion detection models are developed and validated using public benchmark datasets, which may not reflect the operational characteristics, traffic behavior, and threat patterns of real institutional networks. In the case of Umalusi, there is currently no anomaly detection model customized and validated using Umalusi-specific network traffic, creating a practical gap in deployable cybersecurity capability. This study proposes a hybrid machine learning framework tailored to support accurate, efficient, and operationally relevant anomaly detection. Using knowledge discovery in databases (KDD) process, network traffic data were collected and pre-processed through normalization, label encoding, missing value treatment, and dimensionality reduction using principal component analysis (PCA). The 16 hybrid models integrating unsupervised anomaly detection with supervised classification were implemented and comparatively evaluated. Experimental findings indicate that the density-based spatial clustering of applications databasescan (DBSCAN) + random forest (RF) model achieved 99.92% accuracy while maintaining a low false positive (FP) cost, making it suitable for a security operations centre (SOC). In addition, a Flask-based web application was developed to enable real-time deployment by sniffing live network traffic, executing inference, and persisting results in an SQLite database.

Corresponding Author:

Bukohwo Michael Esiefarienrhe
Department of Computer Science and Information Systems, North-West University
Mmabatho Unit 5, Mahikeng, South Africa
Email: Michael.Esiefarienrhe@nwu.ac.za

1. INTRODUCTION

The introduction of artificial intelligence (AI) has made a huge impact on different industries [1], as it has propelled transformative changes, with increased predictive analysis and productivity enhancements in organizational activities [2]. There is a need to explore the adoption and impact of AI, prompting an examination of its effectiveness in the information and communication technology (ICT) security systems of Umalusi. Umalusi is the council responsible for quality assurance in general and further education and training certification for the National Qualification Framework (NQF) levels 1-4 in South Africa.

As corporate networks are increasingly exposed to sophisticated and evolving cyber threats, the cybersecurity systems that protect these networks need to improve, as the techniques used to penetrate them are constantly evolving. Despite the availability of intrusion detection systems (IDS), the network environment at Umalusi is still exposed to abnormal and malicious traffic due to limitations in existing detection methodologies. Existing IDS technologies struggle to keep pace with the complexity and severity

of cyber-attacks, leading to more false alarms and slower detection rate [3] Some of the challenges that exist in IDS as highlighted by [4], [5] are false alarm rate, low detection rate, unbalanced dataset, and response time. AI-based anomaly detection systems are advanced ICT security systems integrated with AI technologies designed to protect the network infrastructure [5].

This research aims to design, implement, and evaluate various sequential hybrid anomaly detection models that integrate unsupervised and supervised machine learning (ML) algorithms to enhance accuracy, precision, and reliability when detecting suspicious behaviors and anomalies within the network infrastructure at Umalusi. Suspicious and abnormal behaviours refer to activities and transactions within the network infrastructure that are outside normal traffic or transactions, such as high network traffic flow (in or out), unusual traffic destinations, and unusual traffic through a particular protocol. The unsupervised ML algorithms used in this study are isolation forest (IF), one class support vector machine (OCSVM), local outlier factor (LOF), and density-based spatial clustering of applications with noise (DBSCAN), combined with four supervised ML algorithms: random forest (RF), decision tree (DT), Naïve Bayes (NB), and K-nearest neighbor (KNN). as these models have proven to have good performance models for anomaly detection from literature. The findings of this research contribute toward developing a practical and deployable anomaly detection approach that supports security operations centre (SOC) monitoring and decision-making in real-world network environments.

Extensive research has been conducted on the application of AI in anomaly detection. In this section, a review is presented of various studies focusing on hybrid anomaly detection models within network environments. Table 1 presents different studies that are examined to explore the models, the ML algorithms, and how they performed on various datasets utilized during experimentation.

The structure of this paper is organized into four sections. The methodology applied in this paper is described in section 2. Different sequential hybrid models, combining unsupervised anomaly detection with supervised classifiers, were trained and evaluated to determine the highest performing model. Section 3 of this paper presents the results of the experiments and discussions of their performance. Section 4 presents the findings and future work to enhance network security using ML.

Table 1. Anomaly detection using hybrid ML

Ref.	Year	Summary
[6]	2024	Their study compared the performance of multiple ML algorithms (supervised, ensemble, and neural models) for IDS, including KNN, SVM, DT, NB, and RF. The authors proposed a hybrid model that utilizes RF and SVM algorithms to leverage the strengths of both algorithms.
[7]	2024	They conducted a study to improve early detection of cyber-attacks by developing a novel hybrid anomaly detection model that integrated DBSCAN and IF. Their study aimed to reduce false positives (FP) and enhance detection accuracy in the network.
[8]	2024	They conducted a study to develop and evaluate hybrid ML models that combine supervised and unsupervised learning approaches to enhance network intrusion detection. Their study aimed to overcome limitations of traditional models by leveraging the strengths of both unsupervised and supervised ML algorithms.
[9]	2023	They conducted a study aimed at evaluating the effectiveness of IF (an unsupervised model) and SVM (a supervised model) for anomaly-based IDS (AIDS). The objective of their study was to address challenges in detecting abnormal network behavior in large-scale data and improve detection accuracy.
[10]	2025	They conducted a study aimed at developing a predictive maintenance system for environmental sensors by detecting anomalies in unlabeled IoT sensor data.
[11]	2022	They conducted a study aimed at developing a hybrid ML model for anomaly detection in network security. The objective of the study was to improve detection accuracy, reduce FP rates, and enhance real-time processing efficiency for IDS.

2. METHOD

This study followed a knowledge discovery in databases (KDD) process to implement a sequential hybrid ML anomaly detection and classification model. KDD process applies the following critical phases to ensure efficient and effective detection: data collection, data preprocessing, data reduction, data mining, and evaluation. This section describes how each of the five steps of the KDD process was applied in this research study, using the methodology depicted in Figure 1. The model with high performance was selected after the evaluation phase. Python was used to implement all the models experimented with.

2.1. Data collection

This study uses the dataset compiled from raw network traffic from the Umalusi network environment. The data was collected directly from live network interfaces and reflects real operational traffic conditions, enabling a realistic evaluation of anomaly detection models. The dataset encompasses two categories of features: core packet attributes, directly captured from each packet, and derived features,

engineered during preprocessing to enhance classification capabilities. Table 2 shows an overview of the dataset features and their descriptions.

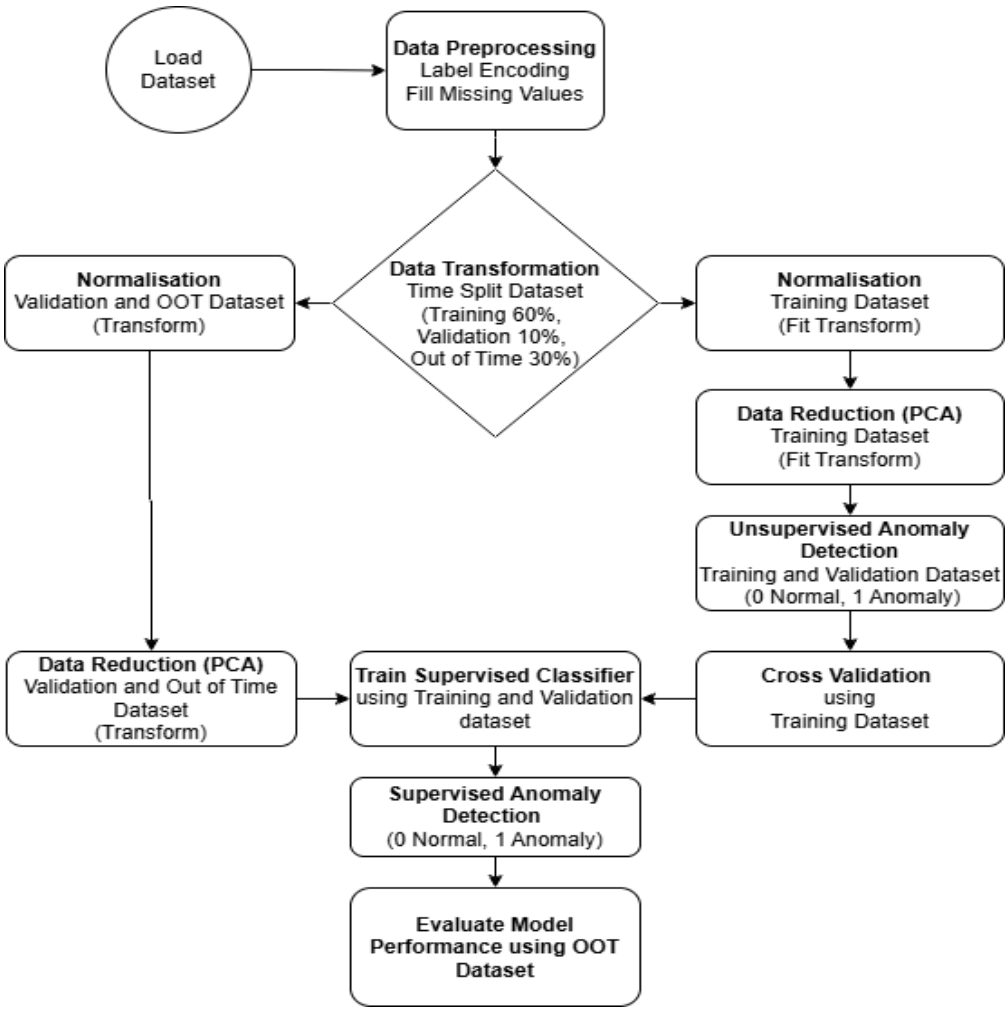

Figure 1. Hybrid anomaly detection methodology

Table 2. Dataset features and their descriptions

Attribute	Feature	Description
Core packet attributes	packet_number	Sequential packet index
	timestamp	Packet arrival time
	packet_size	Total packet length in bytes
	source_port	Source transport port
	dest_port	Destination transport port
	protocol	Internet protocol (IP) identifier
	source_ip	Source IP Address
	dest_ip	Destination IP Address
	cumulative_bytes	Running sum of packet sizes per capture window
Derived attributes	byte_delta	Change in cumulative bytes between consecutive packets
	packet_rate	packets per second
	byte_rate	bytes per second
	duration	Time difference between consecutive packets

2.2. Pre-processing

During the preprocessing stage, irrelevant, duplicate data, and any columns containing data that may not be needed will be removed from the dataset. The dataset may be manually modified where necessary [7]. High-quality data is inclined to yield reliable knowledge compared to low-quality data [12]. This research study applies three data cleaning techniques, label encoding, normalization, and filling of missing values.

The first phase of preprocessing is to apply label encoding. The label encoding method, also known as ordinal encoding, assigns a unique integer to each category from the dataset [13]. For example, different network protocols can each be assigned an integer value, such as 0 for transmission control protocol (TCP), or 1 for user datagram protocol (UDP). The next phase of data preprocessing handles missing values in the dataset. Data cleansing includes methods for dealing with missing values, like imputation (estimating missing values) or eliminating rows or columns that have an excessive number of missing values [14]. The median value of each column will be calculated and used to fill in any missing values.

Dividing the dataset into three subsets, by applying the time-based dataset splitting method, prevents data leakage [15]. The dataset is first sorted in ascending order by timestamp attribute, ensuring that earlier network events precede the later ones. Once ordered, the dataset is divided into three consecutive subsets. 60% for the training set, 10% for the validation set, and 30% for the out-of-time set, as illustrated in Figure 2.

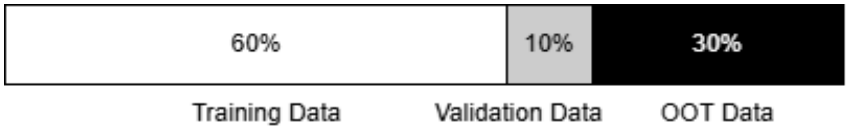

Figure 2. Time-based data split

The last phase ensures the normalization of numerical data to ensure weighting in the ML models. Normalization is achieved by applying max-min scaling on (1) [16]:

$$x_{scaled} = \frac{x_i - x_{min}}{x_{max} - x_{min}} \quad (1)$$

Where, x_{scaled} refers to the normalized feature value, will always lie between 0 and 1. x_i refers to the original feature value, x_{min} is the minimum feature value, x_{max} is the maximum feature value, the largest observed.

2.3. Data reduction

Principal component analysis (PCA) converts the high-dimensional feature space into a lower-dimensional feature space while retaining as much variance (information) as possible [17]. This helps simplify models, reduce computational load, and enhance the classification performance of the AI models. This step is essential for optimizing the application of data mining algorithms, as it reduces computation time and/or enhances their performance [18]. It achieves this by projecting the data onto a smaller set of uncorrelated principal components that capture the maximum variance in the original dataset impacted [19]. PCA functions by first calculating the covariance matrix of the dataset and then carrying out eigenvalue decomposition. The leading eigenvalues (λ), along with their associated eigenvectors (v), serve as the foundation for transforming the data into a lower-dimensional space. Consequently, the new coordinates are derived from these eigenvectors, which identify the directions of maximum variance within the dataset as shown in (2). In this case, $S_v = \lambda_v$ and N represents the number of observations.

$$S = \frac{\sum_{i=1}^{N}(y_i - \mu)(y_i - \mu)^T}{N} \quad (2)$$

2.4. Data mining

The primary purpose of this study is to design, implement, and evaluate a hybrid anomaly detection model that integrates unsupervised and supervised ML algorithms in order to enhance accuracy, precision, and reliability when detecting anomalies within the network infrastructure at Umalusi. The 16 sequential hybrid anomaly detection models were developed, each combining one of four unsupervised algorithms with one of four supervised algorithms. The combinations of these algorithms formed the following models, as indicated in Table 3. These models are categorized according to their respective unsupervised algorithms, resulting in four categories.

2.4.1. Data imbalance

To overcome the problem of data imbalance from the collected data from the Umalusi network, class weight and synthetic minority oversampling technique (SMOTE) are applied. SMOTE prevents class imbalance, which causes ML classifiers to be biased toward the majority class, reducing their ability to detect

anomalies effectively [20], [21]. SMOTE creates artificial data points for the minority class using (3). For each minority class sample x_i, SMOTE selects one of its k-nearest neighbours x_{nn}, and generates a new synthetic same x_{new} [22]:

$$x_{new} = x_i + \lambda(x_{nn} - x_i) \quad (3)$$

Where, x_i is a feature vector from the minority class, x_{nn} is one of the k-nearest neighbours of x_i, $\lambda \in [0,1]$, a random number, x_{new} is a new synthetic data point.

Table 3. List of models developed

Unsupervised	Supervised	Model name
Isolation forest	DT	IF+DT
	NB	IF+NB
	KNN	IF+KNN
	RF	IF+RF
One-class SVM	DT	OCSVM+DT
	NB	OCSVM+NB
	KNN	OCSVM+KNN
	RF	OCSVM+RF
Local outlier factor	DT	LOF+DT
	NB	LOF+NB
	KNN	LOF+KNN
	RF	LOF+RF
DBSCAN	DT	DBSCAN+DT
	NB	DBSCAN+NB
	KNN	DBSCAN+KNN
	RF	DBSCAN+RF

2.5. Evaluation

A confusion matrix, also known as an error matrix, is a table that summarizes the performance of a prediction or classification model [23]. The confusion matrix in Table 4 provides key variables, namely, true positive (TP), true negative (TN), FP, and false negative (FN). The (4)-(7) illustrate how the accuracy, recall, precision, and F1-score were calculated, respectively.

Table 4. Confusion matrix

Confusion matrix		Predicted class	
		Normal	Anomaly
Actual class	Normal	TN	FP
	Anomaly	FN	TP

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$F1\ Score = 2 * \frac{Recall*Precision}{Recall+Precision} \quad (7)$$

3. RESULTS AND DISCUSSION

All the models experimented with were executed on a computer configured with the following hardware and software specifications: 64-bit Windows 11 operating system, Intel i5 processor, 470 GB solid state drive (SSD) storage, 16 GB of random access memory (RAM), Ethernet adapter that supports 1.0 Gigabits per second (Gbps). Python version 3.13 was utilised for ML implementations, operating on Visual Studio Code version 1.102.3 (user setup). The Python libraries required for this experiment are scikit-learn, pandas, numpy, imblearn, flask, and joblib.

3.1. Model performance comparison

The 16 hybrid anomaly detection models, combining unsupervised and supervised ML algorithms, were evaluated using network traffic data collected from the Umalusi network environment. The objective was to identify the most effective model for SOC deployment based on detection performance, stability, ranking capability, and FP cost. Each model was assessed under three class imbalance handling strategies: SMOTE oversampling, supervised class weighting, and a baseline without class balancing. The unsupervised algorithms evaluated were LOF, DBSCAN, IF, and OCSVM. Each unsupervised algorithm was paired with each of four supervised classifiers: RF, DT, NB, and KNN, forming four distinct model categories. The highest-performing hybrid model from each unsupervised category was selected and compared to determine the most suitable model for SOC. Table 5 presents the top-performing models from each category.

Table 5. Highest performers from each category

Model name	Precision	Recall	F1-score	Accuracy (%)
IF+RF	89	87	85	87.29
OCSVM+RF	96	94	95	93.92
LOF+DT	100	99	99	98.60
DBSCAN+RF	100	100	100	99.92

3.2. Confusion matrix

Figure 3 displays the confusion matrix. The IF+RF model in Figure 3(a) indicates a large number of TN (93,064) and a small number of FP (113), indicating that it is successful at reducing false alerts. However, the model has a very high FN rate, missing 14,507 anomalous traffic. Despite a significant percentage of TN, the model misclassified a large number of anomalies as typical traffic. This leads to low recall for the anomaly class, which is a crucial restriction in cybersecurity settings where undiscovered attacks offer a significant risk. Because of the large number of misclassifications, the model becomes unsuitable for SOC deployment.

The LOF+DT model in Figure 3(b) and the OCSVM+RF model in Figure 3(c) had severe classification biases toward the anomalous class, 113451 and 107434, respectively. This classification would result in excessive alert generating in a SOC environment. LOF+DT produced 0 FP and 7 TN. Despite their high detection rates, these two models are unsuitable for operational deployment due to their inability to maintain balanced and meaningful classification.

The confusion matrix for DBSCAN+RF model, demonstrated in Figure 3(d), indicates that this model properly categorized practically all typical traffic, as seen by extraordinarily high TN and a low FP rate. This equates to a near-zero FP rate, which is ideal in a SOC setting because it decreases alert fatigue and operational overhead. Only 7 normal packets were wrongly identified as anomalous, demonstrating high precision.

3.3. Class balance strategies

Figure 4 compares the different class balancing strategies experimented in this study, SMOTE, class weight, and no class balancing, on the accuracy of the four highest performers from each four categories. All models performed relatively consistently across different class-balancing strategies, except the LOF+DT model. The weight class balance for LOF+DT dropped drastically to 48.5% compared to SMOTE and no balancing at 98.6% and 98.24, respectively. The IF-based models were not sensitive to class balancing; the highest accuracy was achieved on SMOTE oversampling. The variation for IF+RF was 0.61% of accuracy. OCSVM+RF model performed better at no balancing, with an accuracy of 93.92%, while its lowest was 91.77%. DBSCAN-based models achieved near-perfect performance across all three strategies, with a variation of only 0.02%. The highest accuracy was produced by no class balancing at 99.96%.

3.4. FP cost analysis

FP cost served as a critical evaluation metric for model selection in this study. Since each false alarm in a SOC environment incurs investigation time and resource expenditure, minimizing FP cost was prioritized alongside traditional performance metrics. The cost value of each model is calculated by applying (8), where FP is the number of FP in a model, $t_{analyst}$ refers to the amount of time it takes to investigate one FP, in which 15 minutes was set for testing purposes. r refers to the rate at which the SOC engineer is compensated for per hour. A standard rate of R300.00 has been set for the purpose of testing different models.

$$FP\ Cost = \text{FP} * \left(\frac{t_{analyst}}{60}\right) * r \quad (8)$$

The best performing model in the OCSVM category, OCSVM+RF, produced the highest FP cost, exceeding R137 000,00. This indicates a large volume of normal traffic classified as anomalies. This suggests poor precision and a high operational burden on SOC analysts. In the IF category, IF+RF produced a moderate FP cost of R 8 475.00. DBSCN+RF produced a good FP cost of R525.00, making it operationally ideal for a SOC environment. In the LOF category, LOF+DT model produced the lowest FP cost of R0.00. This implies that this model did not detect any FP during the model evaluation. While it may be cost efficient, this may be a sign of under detection of actual threats. This is supported by LOF+DT matrix confusion in Figure 3(b), where majority of the network traffic classified as anomalies. From this analysis, DBSCAN+RF has the lowest acceptable FP cost, highlighting its superior precision and minimal analysis intervention requirements. This analysis proves that FP cost is a critical evaluation metric for IDS systems deployed in enterprise environments.

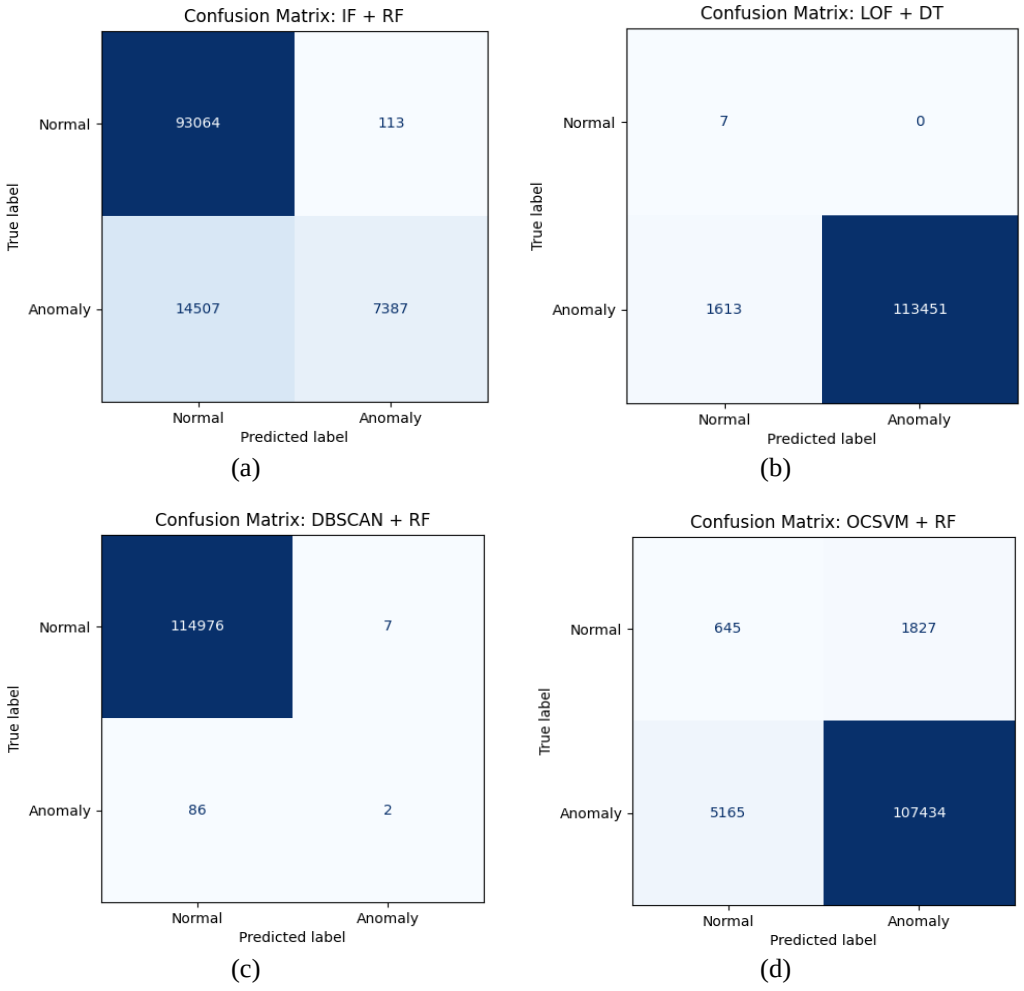

Figure 3. Confusion matrix for high performers: (a) IF+RF, (b) LOF+DT, (c) DBSCAN+RF, and (d) OCSVM+RF

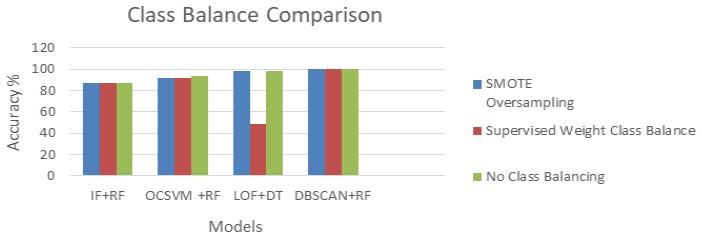

Figure 4. Class balancing strategies

3.5. Discussion on the highest performer

DBSCAN+RF model demonstrates a more robust balance between detection capability, stability, and operational cost, compared to all other models experimented with in this study. This model performed better and proved to be more suited for deployment in real-world SOC environments where traffic distributions are complex, dynamic, and imbalanced. DBSCAN is less sensitive to feature scaling distortions and class imbalances, which are common in real-world network data. This enables DBSCAN to generate cleaner and more separable anomaly labels, which enhances the downstream supervised classifier's learning process.

The hybridization of DBSCAN with RF further amplifies this advantage. The specified DBSCAN parameters ($eps = 0.2$, $min\_samples = 30$) allow the model to capture dense regions of normal network behavior while effectively separating sparse traffic patterns as anomalies, which is consistent with the basic structure and imbalance of real-world network traffic. This allows the RF classifier to focus on discriminative traffic patterns rather than outliers or borderline examples. It leads to better generalization, higher ranking ability, and more consistent performance across validation folds and out-of-time datasets. This model offers the most cost-efficient solution among the evaluated models.

3.6. Model retraining strategy

Model retraining is carried out to ensure that the anomaly detection system stays adaptable to changing network traffic patterns at Umalusi and developing attack behaviors. During retraining, newly collected network traffic data are appended to the existing historical dataset and sorted chronologically to preserve temporal integrity. The supervised model is retrained exclusively on updated normal traffic samples to re-establish an accurate decision boundary, while the unsupervised is retrained using the anomaly labels generated by the updated DBSCAN model. To prevent concept drift and data leakage, retraining adopts a time-based splitting method, with current data set aside for validation and out-of-time (OOT) testing. Metrics such as accuracy, precision, recall, F1-score, and FP cost are used to assess model performance.

3.7. Comparison of the results with existing literature

The results obtained in this research study are comparable to those found in the literature. Table 6 presents a comparison of the performance of this study with that of available studies from literature.

Table 6. Comparison of results with literature

Study	Accuracy (%)
[24]	99.8
[6]	98.98
[7]	97.2
[25]	96
[8]	99
[9]	99
[10]	99.93
[11]	97.39
This study	99.92

3.8. Web application

This study implemented a secure, web-based anomaly detection platform using Python to put the selected high performer (DBSCAN+RF) into practice. The architecture, shown in Figure 5, was created to provide a real-time detection of anomalous network traffic while ensuring scalability, security, and maintainability. The system follows a layered architecture consisting of a presentation layer, application layer, model serving layer, task execution layer, and data persistence layer. The platform consists of four primary components: the client-facing dashboard, the Flask-based web application server, the ML prediction engine (model serving), and the SQLite database for storage. The system is connected to the local area network (LAN) at a strategic point, between the LAN and the internet, to gather all network traffic entering and exiting the LAN. The system follows a layered architecture consisting of the presentation layer, application layer, model serving layer, task execution and queuing, data persistence layer, and security layer. Each layer is discussed in Table 7.

3.9. System performance

To evaluate the performance of the system, we calculated the latency and throughput, for end-to-end batch processing time, which covers data capture, preprocessing, detection, and classification. The latency and throughput are calculated using (9) and (10), respectively, where N is the number of records (packets) processed in a batch, T_s is the inference time in seconds, and T_{ms} is the total inference time in milliseconds.

The limit of each batch count is set to 100, and the timeout is 2 seconds. The system proved to be efficient for real-time network anomaly detection by consistently producing latency at a rate of 50 ms and below, with a throughput of over 2200 packets/sec.

$$Latency\ (ms/record) = \frac{T_{ms}}{N} \quad (9)$$

$$Throughput\ (records/s) = \frac{N}{T_s} \quad (10)$$

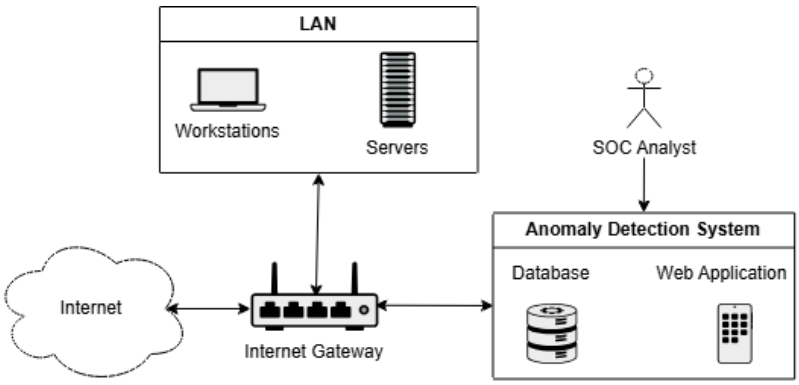

Figure 5. Architectural overview of the anomaly detection system

Table 7. System layers

Layers	Purpose
Presentation layer	The SOC analysts interact with the system through a user interface accessible through a web browser. The interface enables authenticated users to view live network traffic, inspect detected anomalies, and export reports.
Application layer	The application layer is built with Flask framework. The Flask server integrates authentication and authorization controls to protect system resources. All dashboard interfaces and prediction routes are accessible only to authenticated users.
Model serving layer	The machine learning inference subsystem is directly incorporated into the Flask server through model serving with joblib. The trained models and preprocessing components are trained offline and saved as joblib files.
Task execution and queuing	To enable real-time anomaly detection in a live network environment, the system architecture includes a background task execution mechanism that allows for continuous network traffic analysis without interfering with the network. A task queue design allows for asynchronous execution of live traffic collecting and anomaly detection. The system takes queued traffic events, performs preprocessing and dimensionality reduction, and runs the anomaly detection and classification algorithms.
Data storage layer	The storage layer is implemented using SQLite, which provides a lightweight database solution for storing operational detection outputs. SQLite was selected due to its simplicity, minimal configuration requirements, and suitability for small to medium-scale deployments.
Security layer	Due to network monitoring data and anomaly detection results are important, the platform includes security controls that provide confidentiality and controlled access. The Authentication method for these systems is a secure session-based login that ensures that only confirmed users have access to the system.

3.10. Limitations

Although this study achieved strong anomaly detection and classification performance, several limitations remain. The dataset was collected from a single network environment at Umalusi, this means that traffic patterns, device behaviors, and attack characteristics may differ in other organizations, which may limit generalizability. While the system supports near real-time processing, performance may vary under higher traffic volumes, different network interfaces, or resource-constrained deployments.

4. CONCLUSION

This study successfully designed, implemented, and evaluated 16 hybrid ML models for anomaly detection in the Umalusi network environment, following the KDD framework. The models integrated unsupervised and supervised learning techniques, leveraging preprocessing techniques such as normalization,

label encoding, and SMOTE to address class imbalance. Among the tested models, DBSCAN-based and LOF-based hybrid models, particularly DBSCAN+KNN and DBSCAN+RF, demonstrated superior performance, achieving near-perfect accuracy of 99.87% and 99.92%, respectively. These findings highlight the effectiveness of DBSCAN combined with robust classifiers (KNN, RF) in detecting network anomalies with high reliability.

The success of DBSCAN-based models suggests that density-based approaches are highly effective for real-world network traffic analysis, outperforming traditional standalone and some hybrid models reported in prior research. The results align with the existing literature, reinforcing the potential of hybrid ML models in cybersecurity. Future research work may focus on testing and validating the proposed models for scalable deployment in a cloud-based environment. This includes evaluating model performance under varying network loads, assessing latency and throughput in distributed architectures.

ACKNOWLEDGMENTS

We acknowledge the supported given to the researchers by the Umalusi, South Africa and permission to use their network data for the simulation aspects of this research.

FUNDING INFORMATION

This work was supported from the funding provided for postgraduate bursary from the North-West University, South Africa and the Faculty of Natural and Agricultural Sciences postgraduate bursary of the North-West University, South Africa.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Victor Mathebula	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓			
Bukohwo Michael Esiefarienrhe	✓	✓			✓	✓	✓			✓	✓	✓	✓	✓

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

ETHICAL APPROVAL

Although no animal or human was used in this research, but ethical approval was obtained for the research as it is part of a study leading to the award of a master's degree in computer science and information system from the North-West University, South Africa. The Faculty of Natural and Agricultural Science Ethics Committee gave ethical approval as a "No risk" research with ethics certificate Number NWU-01363-24-A9.

DATA AVAILABILITY

The data used in this study were obtained with permission from Umalusi Council (Council for Quality Assurance in General and Further Education and Training, South Africa) and all personal and identifying information were anonymised. Thus, the data is not publicly available, and permission for use should be obtained from Umalusi Council, South Africa.

REFERENCES

[1] M. Zafari, J. S. Bazargani, A. Sadeghi-Niaraki, and S. M. Choi, "Artificial Intelligence Applications in K-12 Education: A Systematic Literature Review," *IEEE Access*, vol. 10, pp. 61905–61921, 2022, doi: 10.1109/ACCESS.2022.3179356.

[2] S. Na, S. Heo, S. Han, Y. Shin, and Y. Roh, "Acceptance Model of Artificial Intelligence (AI)-Based Technologies in Construction Firms: Applying the Technology Acceptance Model (TAM) in Combination with the Technology–Organisation–Environment (TOE) Framework," *Buildings*, vol. 12, no. 2, 2022, doi: 10.3390/buildings12020090.

[3] S. Anand, B. M. Prabodh, and N. A. S. Vinoth, "Real-Time Anomaly Detection Without False Positives Using Genetic Algorithm," in *Proceedings - 2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks, ICICV 2024*, 2024, pp. 552–556. doi: 10.1109/ICICV62344.2024.00093.

[4] V. P. Gandi, N. S. L. Jatla, G. Sadhineni, S. Geddamuri, G. K. Chaitanya, and A. K. Velmurugan, "A Comparative Study of AI Algorithms for Anomaly-based Intrusion Detection," in *Proceedings - 7th International Conference on Computing Methodologies and Communication, ICCMC 2023*, 2023, pp. 530–534. doi: 10.1109/ICCMC56507.2023.10084186.

[5] K. Abhinav and V. Kumar, "A Hybrid Intrusion Detection System Using Machine Learning and Deep Learning," in *2025 3rd International Conference on Communication, Security, and Artificial Intelligence, ICCSAI 2025*, 2025, pp. 1–7. doi: 10.1109/ICCSAI64074.2025.11064178.

[6] V. Sharma and D. J. Shah, "A Novel Approach to Intrusion Detection Systems Using Hybrid Machine Learning Techniques," in *2024 International Conference on Artificial Intelligence and Quantum Computation-Based Sensor Applications, ICAIQSA 2024 - Proceedings*, 2024. doi: 10.1109/ICAIQSA64000.2024.10882184.

[7] M. Nalini, B. Yamini, C. Ambhika, and R. Siva Subramanian, "Enhancing early attack detection: novel hybrid density-based isolation forest for improved anomaly detection," *International Journal of Machine Learning and Cybernetics*, vol. 16, no. 5, pp. 3429–3447, 2025, doi: 10.1007/s13042-024-02460-5.

[8] S. Agrawal, G. K. Gupta, P. K. Gopalakrishna, V. S. Balasubramaniam, L. Goel, and S. Mahadik, "Hybrid Machine Learning Models: Combining Strengths of Supervised and Unsupervised Learning Approaches," in *Proceedings of International Conference on Contemporary Computing and Informatics, IC3I 2024*, 2024, pp. 1056–1061. doi: 10.1109/IC3I61595.2024.10829140.

[9] K. Shanthi and R. Maruthi, "Machine Learning Approach for Anomaly-Based Intrusion Detection Systems Using Isolation Forest Model and Support Vector Machine," in *Proceedings of the 5th International Conference on Inventive Research in Computing Applications, ICIRCA 2023*, 2023, pp. 136–139. doi: 10.1109/ICIRCA57980.2023.10220620.

[10] S. Potharaju, R. K. Tirandasu, S. N. Tambe, D. B. Jadhav, D. A. Kumar, and S. S. Amiripalli, "A two-step machine learning approach for predictive maintenance and anomaly detection in environmental sensor systems," *MethodsX*, vol. 14, 2025, doi: 10.1016/j.mex.2025.103181.

[11] S. Gadal, R. Mokhtar, M. Abdelhaq, R. Alsaqour, E. S. Ali, and R. Saeed, "Machine Learning-Based Anomaly Detection Using K-Mean Array and Sequential Minimal Optimization," *Electronics (Switzerland)*, vol. 11, no. 14, 2022, doi: 10.3390/electronics11142158.

[12] H. M. Safhi, B. Frikh, and B. Ouhbi, "Assessing reliability of Big Data Knowledge Discovery process," *Procedia Computer Science*, vol. 148, pp. 30–36, 2019, doi: 10.1016/j.procs.2019.01.005.

[13] M. K. Dahouda and I. Joe, "A Deep-Learned Embedding Technique for Categorical Features Encoding," *IEEE Access*, vol. 9, pp. 114381–114391, 2021, doi: 10.1109/ACCESS.2021.3104357.

[14] S. M. Siregar, Y. Purwanto, and S. A. Wibowo, "Enhancing Network Anomaly Detection with Optimized One-Class SVM (OCSVM)," in *2023 3rd International Conference on Intelligent Cybernetics Technology and Applications, ICICyTA 2023*, 2023, pp. 84–88. doi: 10.1109/ICICyTA60173.2023.10428830.

[15] Y. Lyu, H. Li, M. Sayagh, Z. M. Jiang, and A. E. Hassan, "An Empirical Study of the Impact of Data Splitting Decisions on the Performance of AIOps Solutions," *ACM Transactions on Software Engineering and Methodology*, vol. 30, no. 4, 2021, doi: 10.1145/3447876.

[16] A. Anjum, P. R. Subramanian, R. Stalinbabu, D. Kothapeta, K. S. Sheela, and B. Jegajothi, "Detecting Zero-Day Attacks using Advanced Anomaly Detection in Network Traffic," in *Proceedings of 5th International Conference on Pervasive Computing and Social Networking, ICPCSN 2025*, 2025, pp. 1247–1253. doi: 10.1109/ICPCSN65854.2025.11034868.

[17] F. Nabi and X. Zhou, "Enhancing intrusion detection systems through dimensionality reduction: A comparative study of machine learning techniques for cyber security," *Cyber Security and Applications*, vol. 2, 2024, doi: 10.1016/j.csa.2023.100033.

[18] B. Molina-Coronado, U. Mori, A. Mendiburu, and J. Miguel-Alonso, "Survey of Network Intrusion Detection Methods from the Perspective of the Knowledge Discovery in Databases Process," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2451–2479, 2020, doi: 10.1109/TNSM.2020.3016246.

[19] B. Subba, S. Biswas, and S. Karmakar, "Enhancing performance of anomaly based intrusion detection systems through dimensionality reduction using principal component analysis," in *2016 IEEE International Conference on Advanced Networks and Telecommunications Systems, ANTS 2016*, 2017. doi: 10.1109/ANTS.2016.7947776.

[20] B. Bala and S. Behal, "AI techniques for IoT-based DDoS attack detection: Taxonomies, comprehensive review and research challenges," *Computer Science Review*, vol. 52, 2024, doi: 10.1016/j.cosrev.2024.100631.

[21] K. Negi, G. P. Kumar, G. Raj, S. Sahana, and V. Jain, "Degree of Accuracy in Credit Card Fraud Detection Using Local Outlier Factor and Isolation Forest Algorithm," *2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2022, pp. 240-245, doi: 10.1109/Confluence52989.2022.9734123.

[22] I. Dey and V. Pratap, "A Comparative Study of SMOTE, Borderline-SMOTE, and ADASYN Oversampling Techniques using Different Classifiers," in *Proceedings - 2023 3rd International Conference on Smart Data Intelligence, ICSMDI 2023*, 2023, pp. 294–302. doi: 10.1109/ICSMDI57622.2023.00060.

[23] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, and M. Xu, "A Survey on Machine Learning Techniques for Cyber Security in the Last Decade," *IEEE Access*, vol. 8, pp. 222310–222354, 2020, doi: 10.1109/ACCESS.2020.3041951.

[24] T. P. Jayesh, K. Pandiaraj, A. Paul, R. R. Chandran, and P. P. Menon, "A Hybrid Machine Learning Approach to Anomaly Detection in Industrial IoT," in *ACCESS 2023 - 2023 3rd International Conference on Advances in Computing, Communication, Embedded and Secure Systems*, 2023, pp. 32–36. doi: 10.1109/ACCESS57397.2023.10199711.

[25] P. Maturure, A. Ali, and A. Gegov, "'Hybrid Machine Learning Model for Phishing Detection,'" in *2024 IEEE 12th International Conference on Intelligent Systems (IS)*, 2024, no. 2024. doi: 10.1109/IS61756.2024.10705257.

BIOGRAPHIES OF AUTHORS

Victor Mathebula received his B.Tech. in Communication Networks from Tshwane University of Technology in 2016. He obtained his B.Sc. Hons in Computing from the University of South Africa in 2022. He has worked for various South African Government Institutions, focusing on ICT infrastructure, security, and risk management. Presently, he is pursuing M.Sc. in Computer Science at North-West University, Mahikeng, in South Africa. His research interests include cybersecurity, machine learning, and data mining. He can be contacted at email: victorwaleriver@gmail.com.

Bukohwo Michael Esiefarienrhe received B.Sc. (UNIBEN), M.Sc. (ATBU), Ph.D. (MAUT, Yola) degrees in Computer Science. He is an oracle certified professional and a member of several professional bodies including IEEE, ACM, IITPSA, CPN. He is currently the head of department and professor of Computer Science and Information Systems, North-West University, South Africa. He has published over 70 articles in peer-reviewed journals and has presented several scholarly papers at conferences both locally and internationally. His research interests span AI and machine learning, software engineering, ICT infrastructures and wireless sensors networks. He has graduated over 10 doctoral, over 30 masters, and over 80 honours students in computer science and information systems. He can be contacted at email: Michael.Esiefarienrhe@nwu.ac.za.