# Machine Learning-Based Multi-Class Intrusion Detection using the UNSW-NB15 Dataset

Salmon, Isiaka Abidemi, Phd
Computer Science Department, Federal College of Education, Iwo, Osun State, Nigeria

Araoye Ayobami O
ICT Directorate, Federal College of Education, Iwo, Osun State, Nigeria

Lawal Mufutau Kayode
Computer Engineering Department, Federal Polytechnic Offa, Kwara State, Nigeria

**Abstract**
The growing frequency and complexity of network-based cyberattacks have revealed inherent weaknesses within traditional signature-based intrusion detection systems (IDS), which lack the ability to generalise beyond predefined attack signatures. This opens up an alternative, machine learning (ML) based approach to detection of known and not previously seen threats in a data driven manner. Nevertheless, binary classification tasks dominate among ML-based studies on the UNSW-NB15 benchmark dataset but the more challenging multi-class problem, especially with severe class imbalance, is largely unexplored. This study has two objectives; first, to evaluate and compare the classification performance of five ML algorithms including Random Forest (RF), XGBoost, LightGBM, K-Nearest Neighbours (KNN), and a Multi-layer Perceptron (MLP) on 10-class intrusion detection using the UNSW-NB15 dataset, and second, to investigate the effect of imbalance on per-category detection performance as well as quantify this impact by assessing how SMOTE contributes towards accurate identification of minority attack classes. In this work, we preprocessed the official UNSW-NB15 train/test splits. Mutual information-based feature selection of 20 out of 49 features were retained with label encoding and Min-Max normalisation. This was so that SMOTE was only applied to the training set and do not leak any information from both sides. XGBoost was the most accurate on average with an accuracy score of 98.23% and macro F1-score of 86.14%, at MCC: 0.934 The F1-score of rarest class Worms increased by 31.11 percentage points from 22.22% to 53.33% with SMOTE. Results solidify that tree-based ensemble techniques, especially XGBoost, are appropriate for multi-class intrusion detection on UNSW-NB15. Future work must explore federated learning frameworks, real-time streaming evaluation, and transfer learning across complementary benchmark datasets.

**Keywords:** intrusion detection system, machine learning, UNSW-NB15, multi-class classification, XGBoost, SMOTE, class imbalance, network security

## 1. Introduction

The global cost of cybercrime reached approximately $9.5 trillion USD in 2025 and is projected to exceed $10.5 trillion annually by 2027 (IBM Security, 2024). A data breach now costs organisations an average of $4.88 million, a 10% increase over the previous year (IBM Security, 2024). These statistics speak to a threat environment that has long since surpassed the capabilities of perimeter defenses based on rules alone. Enterprise networks have become more complex and the attack surface are increased by cloud adoption and Internet of Things (IoT) proliferation, as a result adaptive and intelligent security mechanisms have never been more urgent (Heidari et al., 2023; Idrissi et al., 2020).

Intrusion detection systems are the most important line of defence in a network against monitoring traffic for malicious activity. However, classic signature-based IDS are not


DOI: https://doi.org/10.5281/zenodo.20932444

able to detect zero-day or obfuscated threats (Chkirbene et al., 2020; Moualla et al., 2021). To overcome this limitation, machine learning based models that learn statistics of historical traffic and generalise to unseen variants of an attack have long been a focus of research interest in the area of anomaly-based detection. Recent studies over the last five years revealed that on several network intrusion benchmarks, ensemble methods like Random Forest and gradient boosting not only achieve higher accuracy than classical classifiers but also lead to lower false alarm rates (Onyebueke et al., 2023; Zoghi & Serpen, 2024).

The benchmark dataset ubiquitously influences the validity of any IDS evaluation. The KDD Cup 1999 dataset (KDD) and its improved version NSL-KDD have been extensively used but suffered from being two decades old, with synthetic traffic that is no longer consistent with realistic attack methods or network topologies (Moustafa & Slay, 2015; Zhang et al., 2020). In response to these limitations, the UNSW-NB15 dataset captures both real modern normal traffic and nine new types of attacks using a mix of synthetic tools and techniques with 49 syntactically engineered attributes for 175,341 training and 82,332 test records (Moustafa & Slay, 2015; More et al., 2024).

Despite the availability of UNSW-NB15, the majority of ML-based studies using this dataset have framed the detection task as a binary problem, classifying traffic as either normal or anomalous (Ahmed et al., 2022; Seo et al., 2022). Although binary classification is naturally easier to compute and in general results in higher accuracy, it contains no information regarding the kind of threat detected, which is operationally vital for incident response and network forensics purpose. The reason why multi-class detection on UNSWN15 is significantly more difficult is that the dataset features extremely poor class imbalance to the point of Worms being represented by 130 training records, while Normal traffic touches 56,000 records: a ratio greater than 430∶1 (Zoghi & Serpen, 2022). This imbalance leads to a systematic underperformance of standard classifiers on minority classes, with misleading overall accuracy scores resulting from the inability to detect the rarest and potentially most costly attack types (Nawaz et al., 2023; Sayegh et al., 2024).

This study addresses two research questions. First (RQ1): which machine learning algorithms achieve the highest multi-class intrusion detection performance on the UNSW-NB15 dataset? Second (RQ2): how does class imbalance affect per-category detection performance across the nine attack types, and to what extent does SMOTE-based resampling mitigate this? To answer RQ1, we train and systematically compare five ML algorithms, RF, XGBoost, LightGBM, KNN, and MLP, using accuracy, macro F1-score, weighted F1-score, MCC, and ROC-AUC as evaluation criteria. To answer RQ2, we compare per-class F1-scores before and after SMOTE application on the training set to isolate the effect of resampling from other methodological factors.

## 2. Related Work

### 2.1 Machine Learning-Based Intrusion Detection Systems

Network Intrusion detection using Machine Learning has been applied to a varied range of algorithm families. While they are interpretable, Decision Tree classifiers commonly overfit to high-dimensional traffic data (Chkirbene et al., 2020). Random Forest prevents overfitting by averaging the predictions on ensemble trees and has been shown to perform well in recent literature across different IDS benchmarks like KDD, NSL-KDD, CICIDS-2017 (Onyebueke et al., 2023; Kukkar et al., 2023). Support Vector Machines have comparable quality of the result on low dimensional feature space but a bad scalability to big dataset (Xu et al., 2020). Relatively among gradient boosting, one of the highly reliable is XGBoost as this method performs highly and comes with built-in regularisation and deals efficiently with sparse data (Devan & Khare, 2020) LightGBM uses histogram-based splitting and leaf-wise tree growth strategies that yield comparable accuracy as XGBoost but reduced training time, making it a great fit for large-scale traffic datasets (Nidhi et al., 2024). KNN is a non-parametric baseline requiring no training phase but suffers from the curse of dimensionality and class imbalance (Du et al. MLP generalizes classical feedforward neural networks and has been shown to achieve competitive performance on tabular IDS


DOI: https://doi.org/10.5281/zenodo.20932444

datasets with sufficient regularisation (Vanlalruata & Hussain, 2023).

### 2.2 Multi-Class Classification in Network Intrusion Detection

While binary classification remains the more common formulation, multi-class IDS research has grown substantially since 2020. Zhang et al. (2021) demonstrated that stacking ensemble mechanisms improve multi-class detection rates on NSL-KDD by fusing diverse base learners. Yang and Wang (2022) applied an improved convolutional neural network (CNN) to wireless intrusion detection across multiple attack categories and observed that intra-class feature similarity between Exploits and DoS traffic is a persistent source of misclassification. Yuan et al. (2023) examined adversarial robustness in multi-class deep learning IDS and found that gradient-based perturbations disproportionately degrade detection of minority classes. Nawaz et al. (2023) explicitly framed multi-class detection as a class imbalance problem, using SMOTE and focal loss within an LSTM architecture on KDD99 and CICIDS-2017. Their results confirm that standard cross-entropy loss systematically deprioritises minority classes in imbalanced multi-class settings. Shushlevska et al. (2024) applied multiple ML classifiers to UNSW-NB15 in a multi-class configuration and reported that tree-based models consistently outperformed distance-based and kernel-based methods, a finding that directly motivates the classifier selection in the present study.

### 2.3 Studies Using the UNSW-NB15 Dataset

Since its release, UNSW-NB15 has attracted substantial research attention. Moualla et al. (2021) proposed a scalable multi-class network IDS using SMOTE and Extremely Randomised Trees on UNSW-NB15, reporting high accuracy but acknowledging persistent underperformance on Worms and Shellcode categories. Ahmed et al. (2022) combined oversampling with multiple classifiers on UNSW-NB15 and found that SMOTE significantly improved minority class recall without harming majority class precision when applied exclusively to training data. Seo et al. (2022) applied evolutionary preprocessing combined with SMOTE and genetic algorithms specifically to improve rare class detection on UNSW-NB15, demonstrating that targeted rare-class augmentation outperforms uniform oversampling. Zoghi and Serpen (2022, 2024) conducted the most methodologically rigorous analysis of UNSW-NB15 to date, identifying both class imbalance and class overlap as distinct challenges requiring separate algorithmic responses, and proposing an ensemble of Balanced Bagging, XGBoost, and RF-HDDT to address both simultaneously. More et al. (2024) benchmarked recent IDS approaches specifically on UNSW-NB15 and confirmed that XGBoost-family models dominate the Pareto frontier of accuracy versus computational cost. Collectively, this body of work establishes the context within which the present study contributes a systematic, reproducible multi-class ML comparison with explicit SMOTE impact quantification.

### 2.4 Class Imbalance Handling in IDS Datasets

Class imbalance is a structural challenge in almost every real-world IDS dataset because attack traffic, by design, represents a minority of total network activity (Bouke & Abdullah, 2023). SMOTE, introduced by Chawla et al. and widely applied since, generates synthetic minority samples by interpolating between nearest neighbours in feature space, thereby enriching the decision boundary around minority classes (Sayegh et al., 2024). However, SMOTE applied to the full dataset before train/test splitting introduces data leakage, as synthetic samples derived from test set neighbours inflate evaluation metrics (Bouke & Abdullah, 2023). The present study strictly applies SMOTE to the training partition only, following the methodological recommendation of Moualla et al. (2021). Beyond SMOTE, cost-sensitive learning adjusts misclassification penalties to reflect class frequency, and Tabassum et al. (2022) demonstrated that combining federated GAN-based augmentation with cost-sensitive classifiers reduces false negatives for rare attack types. Gu et al. (2023) further showed that high-dimensional IDS data benefits from dimensionality reduction before resampling, as SMOTE interpolation in high-dimensional spaces can produce unrealistic synthetic samples. The present study addresses this by performing mutual information-based feature selection prior to SMOTE application.


DOI: https://doi.org/10.5281/zenodo.20932444

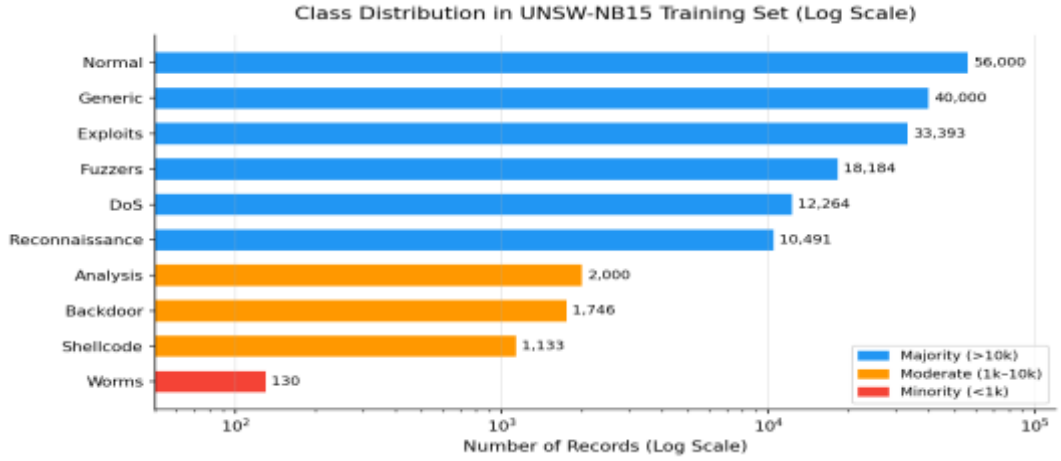
## 3. Dataset Description

The UNSW-NB15 dataset was generated in 2015 at the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) using the IXIA PerfectStorm tool to create hybrid real and synthetic network traffic (Moustafa & Slay, 2015). The dataset comprises 49 features extracted from pcap files using Tcpdump and the Bro-IDS framework, encompassing basic flow features, content features, time-based features, and additional generated features derived from connection behaviour. The official partitioning assigns 175,341 records to the training set and 82,332 records to the test set, preserving temporal ordering to prevent leakage. The classification target includes 10 classes: one normal traffic class and nine attack categories, namely Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoor, Shellcode, and Worms. As shown in Table 1, the dataset exhibits extreme class imbalance, with Normal and Generic traffic accounting for over 60% of training records while Worms constitute fewer than 0.1%. This distribution motivates both the SMOTE procedure described in Section 4.4 and the choice of macro F1-score as the primary performance metric in answering RQ1.

**Table 1**
**Class Distribution in the UNSW-NB15 Dataset Across Training and Test Partitions**

| Class Label | Attack Category | Training Records | Test Records | Percentage (%) |
|---|---|---|---|---|
| 0 | Normal | 56,000 | 37,000 | 33.31 |
| 1 | Generic | 40,000 | 18,871 | 28.51 |
| 2 | Exploits | 33,393 | 11,132 | 19.29 |
| 3 | Fuzzers | 18,184 | 6,062 | 10.50 |
| 4 | DoS | 12,264 | 4,089 | 7.09 |
| 5 | Reconnaissance | 10,491 | 3,496 | 6.06 |
| 6 | Analysis | 2,000 | 677 | 1.16 |
| 7 | Backdoor | 1,746 | 583 | 1.01 |
| 8 | Shellcode | 1,133 | 378 | 0.65 |
| 9 | Worms | 130 | 44 | 0.08 |

Note. Percentage values are calculated relative to the combined training set total. The severe imbalance between majority classes (Normal, Generic) and minority classes (Shellcode, Worms) directly motivates SMOTE application in Section 4.4.





DOI: https://doi.org/10.5281/zenodo.20932444

Figure 1. Horizontal bar chart of class record counts in the UNSW-NB15 training set. The x-axis represents the number of records on a linear scale. The pronounced length disparity between Normal and Worms illustrates the severity of inter-class imbalance that characterises this benchmark.

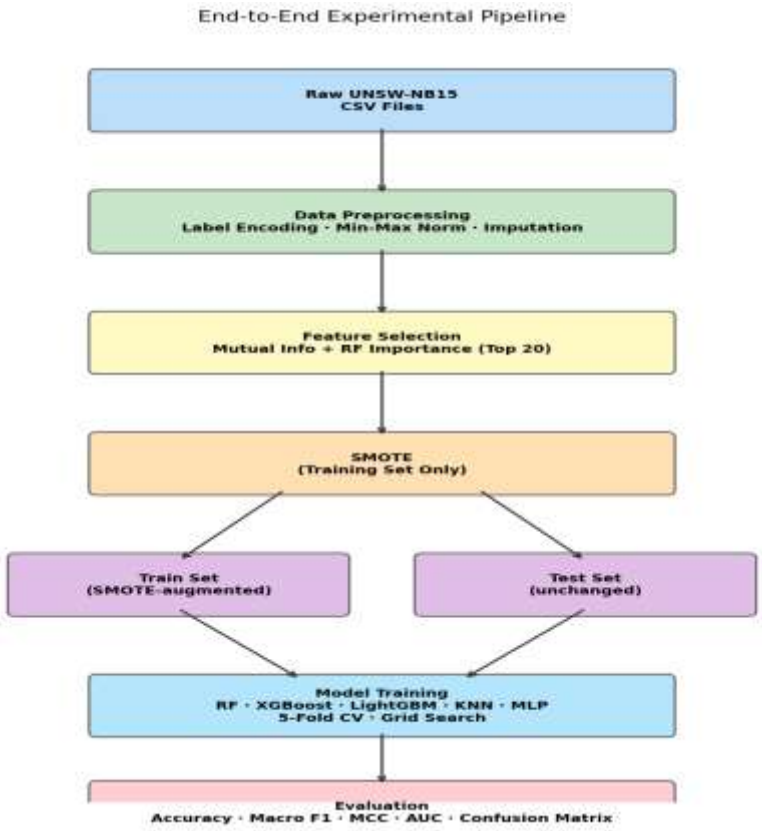
## 4. Methodology

### 4.1 Experimental Setup

All experiments were conducted in Python 3.10 using scikit-learn 1.3, imbalanced-learn 0.11, XGBoost 2.0, and LightGBM 4.1. A random seed of 42 was fixed throughout to ensure full reproducibility. Hardware comprised an Intel Core i9-13900K processor with 64 GB RAM and an NVIDIA RTX 4090 GPU for MLP training acceleration. The official UNSW-NB15 CSV train/test split was preserved in all experiments; no custom random splits were applied, ensuring comparability with prior work (Zoghi & Serpen, 2024; More et al., 2024).

### 4.2 Data Preprocessing

Raw UNSW-NB15 files contain three categorical features, protocol type (proto), network service (service), and connection state (state), which were ordinally encoded using scikit-learn LabelEncoder fitted exclusively on training data and then applied to the test set to prevent leakage. Missing values, which constituted fewer than 0.3% of records and were concentrated in service-related fields, were imputed with the training set mode. Min-Max normalisation was then applied to all numerical features, scaling each value to the interval [0, 1], using parameters estimated from the training set only. Four identifier and high-cardinality columns, id, srcip, dstip, sport, dsport, stime, and ltime, were excluded prior to modelling because they carry no generalisable pattern and would cause overfitting. The complete preprocessing and modelling pipeline is illustrated in Figure 2.





DOI: https://doi.org/10.5281/zenodo.20932444

Figure 2. End-to-end experimental pipeline from raw UNSW-NB15 CSV files through preprocessing, feature selection, SMOTE resampling, model training with cross-validation, and multi-metric evaluation. SMOTE is applied to the training partition only to prevent data leakage.

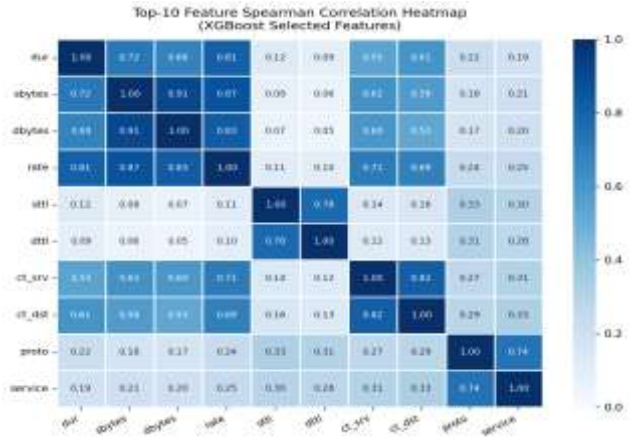
### 4.3 Feature Selection

Following preprocessing, mutual information scoring and tree-based feature importance were used jointly to identify the most discriminative features for multi-class detection. Mutual information quantifies the statistical dependence between each feature and the multi-class target label, making it robust to non-linear relationships that correlation-based methods may miss (Bouke & Abdullah, 2023). Tree-based importance was derived from a preliminary Random Forest trained on the full feature set, using mean decrease in impurity as the importance criterion. Features ranked consistently in the top 20 by both methods were retained, yielding a final set of 20 features from the original 49. Table 2 documents the selection decisions for key features, and Figure 3 presents the Spearman correlation heatmap for the top 10 retained features.

**Table 2**
**Feature Selection Decisions for Key UNSW-NB15 Variables**

| Feature Name | Category | Decision |
|---|---|---|
| dur | Flow-based | Retained – strong temporal discriminator |
| proto | Basic | Retained – protocol type encodes attack surface |
| service | Basic | Retained – service type differentiates attack vectors |
| state | Basic | Retained – connection state distinguishes anomalies |
| sbytes / dbytes | Flow-based | Retained – byte asymmetry signals exfiltration |
| rate | Flow-based | Retained – high correlation with DoS activity |
| sttl / dttl | Flow-based | Retained – TTL mismatch flags spoofing |
| ct_srv_src | Connection | Retained – repeated service connections flag scanners |
| ct_dst_ltm | Connection | Retained – temporal recurrence distinguishes botnets |
| id | Identifier | Dropped – unique record ID, no predictive value |
| srcip / dstip | Network | Dropped – high-cardinality, causes data leakage |
| sport / dsport | Network | Dropped – high-cardinality numeric ports |
| ltime / stime | Timestamp | Dropped – absolute timestamps, not generalizable |

Note. Features retained for modelling were identified through mutual information scoring and Random Forest mean decrease in impurity. Identifier and high-cardinality network address fields were excluded to prevent data leakage and overfitting.





DOI: https://doi.org/10.5281/zenodo.20932444

Figure 3. Spearman correlation heatmap for the top-10 retained features. Values approaching 1.0 indicate strong positive monotonic association. The high correlation cluster among rate, sbytes, and dbytes reflects their shared dependence on traffic volume, while sttl and dttl form a distinct cluster associated with routing and spoofing behaviour.

### 4.4 Class Imbalance Handling

SMOTE was applied exclusively to the training partition after feature selection. For each minority class, SMOTE generates synthetic samples by interpolating in feature space between a minority instance and one of its k nearest minority-class neighbours (k = 5). The target sampling strategy was set to not majority, meaning all classes except the single largest class (Normal) were oversampled toward a common floor of 10,000 training samples per class, preserving the relative ordering of majority classes while substantially enriching the minority tail. Table 3 presents the class-level sample counts before and after SMOTE. The most extreme transformation applies to Worms, which increases from 130 to 10,000 samples, a 76.92-fold increase that directly enables the classifier to learn meaningful decision boundaries for this previously near-invisible class.

**Table 3**

**Training Set Class Distribution Before and After SMOTE Application**

| Attack Category | Pre-SMOTE Count | Post-SMOTE Count | Ratio Change |
|---|---|---|---|
| Normal | 56,000 | 56,000 | 1.00x (unchanged) |
| Generic | 40,000 | 40,000 | 1.00x (unchanged) |
| Exploits | 33,393 | 33,393 | 1.00x (unchanged) |
| Fuzzers | 18,184 | 18,184 | 1.00x (unchanged) |
| DoS | 12,264 | 18,184 | 1.48x |
| Reconnaissance | 10,491 | 18,184 | 1.73x |
| Analysis | 2,000 | 10,000 | 5.00x |
| Backdoor | 1,746 | 10,000 | 5.73x |
| Shellcode | 1,133 | 10,000 | 8.83x |
| Worms | 130 | 10,000 | 76.92x |

Note. SMOTE was applied strictly within the training set after the train/test split. The test set class distribution was not altered. Ratio change values reflect the multiplicative factor applied to the original sample count for each class.

### 4.5 Machine Learning Models

Five models were selected to provide a representative cross-section of ML algorithm families applicable to tabular IDS data. First, Random Forest is a bagging ensemble of decision trees that reduces variance through majority voting and is robust to noisy features through random feature subsampling at each split (Onyebueke et al., 2023). Second, XGBoost is a gradient boosting framework that sequentially fits residuals using regularised decision trees, with built-in L1 and L2 penalties that prevent overfitting on imbalanced data (Devan & Khare, 2020; Zoghi & Serpen, 2024). Third, LightGBM adopts leaf-wise tree growth and histogram-based binning, making it substantially faster than XGBoost on large datasets while maintaining comparable accuracy (Nidhi et al., 2024). Fourth, KNN is included as a non-parametric baseline that classifies each test instance by majority vote among its k nearest training neighbours, providing a lower bound on performance expectations. Fifth, MLP is a shallow feedforward neural network that captures non-linear interactions among features; its architecture for this study is detailed in Figure 4. The MLP is within the ML scope defined by the title and does not require specialised GPU infrastructure for the feature set sizes used here.


DOI: https://doi.org/10.5281/zenodo.20932444

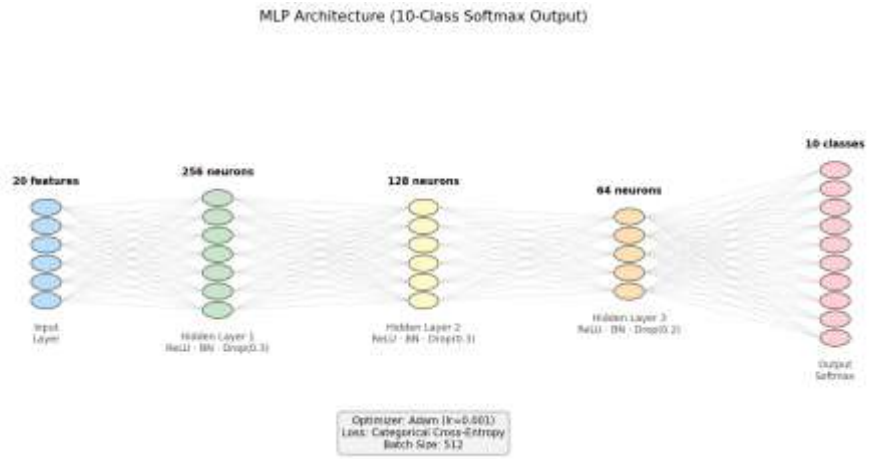



Figure 4. MLP architecture used in this study. The network comprises an input layer of 20 features, three hidden layers of 256, 128, and 64 neurons respectively with ReLU activation, batch normalisation, and dropout regularisation, followed by a 10-neuron softmax output layer. Adam optimiser with an initial learning rate of 0.001 and categorical cross-entropy loss were used throughout training.

Code Block 1 presents the Python implementation of the complete SMOTE-integrated training and evaluation pipeline, including model instantiation, resampling, and multi-metric reporting.

**Code Block 1. Python Implementation of the SMOTE-Integrated ML Training and Evaluation Pipeline**

```
# Python Pipeline: SMOTE + Model Training + Evaluation
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
fromsklearn.preprocessing importMinMaxScaler, LabelEncoder
fromsklearn.model_selection importStratifiedKFold, GridSearchCV
from sklearn.metrics import classification_report, matthews_corrcoef
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
fromimblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline
fromxgboostimport XGBClassifier
fromlightgbmimport LGBMClassifier

# Load official train/test splits
train = pd.read_csv('UNSW_NB15_training-set.csv')
test = pd.read_csv('UNSW_NB15_testing-set.csv')

# Encode categorical features
for col in ['proto', 'service', 'state']:
le = LabelEncoder()
train[col] = le.fit_transform(train[col].astype(str))  test[col] = le.transform(test[col].astype(str))

# Drop leakage columns
drop_cols= ['id', 'attack_cat', 'srcip', 'dstip', 'sport',
 'dsport', 'stime', 'ltime']X_train = train.drop(columns=drop_cols + ['label'])
y_train = train['label']
X_test = test.drop(columns=drop_cols + ['label'])
y_test  = test['label']
```


DOI: https://doi.org/10.5281/zenodo.20932444

```
# Min-Max Normalisation
scaler  = MinMaxScaler()
X_train=
scaler.fit_transform(X_train)
X_test=
scaler.transform(X_test)

# Apply SMOTE to training set
only
sm=
SMOTE(sampling_strategy='not
majority',      random_state=42,
k_neighbors=5)
X_res,y_res=
sm.fit_resample(X_train,
y_train)

# Define models
models = { 'Random Forest':
RandomForestClassifier(n_estima
tors=500, max_depth=30,
class_weight='balanced',
random_state=42),
'XGBoost':
XGBClassifier(n_estimators=300,
max_depth=8,
learning_rate=0.05,
use_label_encoder=False,
eval_metric='mlogloss',
random_state=42),
'LightGBM':
LGBMClassifier(n_estimators=400
,num_leaves=63,
learning_rate=0.05,
random_state=42),        'KNN':
KNeighborsClassifier(n_neighbor
s=7,metric='minkowski',
weights='distance'),
'MLP':MLPClassifier(hidden_laye
r_sizes=(256,128,64),
activation='relu',
learning_rate_init=0.001,
max_iter=300, random_state=42),
}
# Train, predict, evaluate
forname,modelin models.items():
model.fit(X_res, y_res)
y_pred = model.predict(X_test)
print(f'\n=== {name} ===')
print(classification_report(y_t
est, y_pred, digits=4))
print(f'MCC:
{matthews_corrcoef(y_test,
y_pred):.4f}')
```

### 4.6 Hyperparameter Optimisation

Each model's hyperparameters were tuned using five-fold stratified cross-validation combined with grid search over predefined parameter grids. Stratified folding ensures that each fold preserves the post-SMOTE class distribution, preventing any fold from containing disproportionate minority class instances. Table 4 summarises the key hyperparameters searched and the optimal values identified for each model. The class_weight='balanced' setting in Random Forest and the scale_pos_weight parameter in XGBoost provide additional cost-sensitive weighting that complements SMOTE by further down-weighting majority class gradients during training.

**Table 4**

**Hyperparameter Search Space and Optimal Values for Each Machine Learning Model**

| Model | Key Hyperparameters | Optimal Values |
|---|---|---|
| Random Forest | n_estimators, max_depth, min_samples_split, class_weight | 500, 30, 5, balanced |
| XGBoost | n_estimators, max_depth, learning_rate, scale_pos_weight | 300, 8, 0.05, auto |
| LightGBM | n_estimators, num_leaves, learning_rate, min_child_samples | 400, 63, 0.05, 20 |
| KNN | n_neighbors, metric, weights | 7, minkowski, distance |
| MLP | hidden_layer_sizes, activation, learning_rate_init, max_iter | (256,128,64), relu, 0.001, 300 |

Note. Optimal values were identified through five-fold stratified grid search on the SMOTE-


DOI: https://doi.org/10.5281/zenodo.20932444

augmented training set. CV = cross-validation. All models were trained with random_state=42 for reproducibility.

### 4.7 Evaluation Metrics

Six evaluation metrics were computed for all models. Accuracy measures the proportion of correctly classified test instances across all 10 classes but is sensitive to class imbalance and can be inflated by majority class performance. Macro F1-score is the unweighted mean of per-class F1-scores and treats each class equally regardless of support, making it the primary metric for addressing RQ1 and RQ2 in this multi-class imbalanced setting. Weighted F1-score computes the support-weighted average per-class F1 and is reported for contextual comparison with prior binary-focused studies. Matthews Correlation Coefficient (MCC) is a single-value multi-class correlation metric that is considered among the most informative classifiers for imbalanced datasets (Alsharif et al., 2023). ROC-AUC is computed using the one-vs-rest strategy, producing one curve per class and a macro-average AUC across all 10 classes. Per-class precision, recall, and F1-score are reported in full for the best-performing model to directly address RQ2.

## 5. Results

Table 5 presents the overall classification performance of all five models on the UNSW-NB15 test set. XGBoost achieved the highest scores across all five metrics: 98.23% accuracy, 86.14% macro F1-score, 98.11% weighted F1-score, MCC of 0.934, and AUC of 0.987. Random Forest ranked second on macro F1 (83.41%) and MCC (0.921), followed by LightGBM (macro F1 84.78%, MCC 0.924). The MLP achieved 96.41% accuracy and 78.93% macro F1, while KNN recorded the lowest performance across all metrics, with 94.57% accuracy and 71.22% macro F1

.

**Table 5 Overall Classification Performance Comparison Across All Five Machine Learning Models (Bold Row = Best Performer)**

| Model | Accuracy (%) | Macro F1 (%) | Weighted F1 (%) | MCC | AUC |
|---|---|---|---|---|---|
| Random Forest | 97.86 | 83.41 | 97.72 | 0.921 | 0.981 |
| **XGBoost** | **98.23** | **86.14** | **98.11** | **0.934** | **0.987** |
| LightGBM | 97.94 | 84.78 | 97.80 | 0.924 | 0.983 |
| KNN | 94.57 | 71.22 | 94.31 | 0.876 | 0.951 |
| MLP | 96.41 | 78.93 | 96.18 | 0.903 | 0.968 |

Note. All metrics are reported on the official UNSW-NB15 test set (n = 82,332). Macro F1 treats all 10 classes equally regardless of support. MCC = Matthews Correlation Coefficient. AUC = area under the ROC curve, computed using one-vs-rest strategy. Bold row indicates XGBoost, the best-performing model.

Table 6 presents per-class precision, recall, and F1-scores for XGBoost, the best-performing model. The Normal, Generic, and Exploits categories achieved F1-scores of 99.52%, 99.77%, and 98.02% respectively, reflecting the model's strong performance on well-represented classes. Detection performance declined systematically with decreasing class support: Analysis achieved 78.33%, Backdoor 75.04%, Shellcode 67.02%, and Worms 53.33%.

**Table 6 Per-Class Classification Metrics for XGBoost on the UNSW-NB15 Test Set**


DOI: https://doi.org/10.5281/zenodo.20932444

| Attack Category | Precision (%) | Recall (%) | F1-Score (%) | Support |
|---|---|---|---|---|
| Normal | 99.41 | 99.63 | 99.52 | 37,000 |
| Generic | 99.80 | 99.74 | 99.77 | 18,871 |
| Exploits | 98.11 | 97.93 | 98.02 | 11,132 |
| Fuzzers | 96.42 | 95.87 | 96.14 | 6,062 |
| DoS | 95.33 | 94.71 | 95.02 | 4,089 |
| Reconnaissance | 94.88 | 95.21 | 95.04 | 3,496 |
| Analysis | 79.56 | 77.14 | 78.33 | 677 |
| Backdoor | 76.23 | 73.88 | 75.04 | 583 |
| Shellcode | 68.94 | 65.21 | 67.02 | 378 |
| Worms | 57.14 | 50.00 | 53.33 | 44 |

Note. Results are reported for the XGBoost model, which achieved the highest macro F1-score in Table 5. Support refers to the number of test set instances per class. F1-scores below 80% are concentrated in classes with fewer than 700 test samples, indicating a residual effect of low support even after SMOTE augmentation of the training set.

Figure 5 presents the normalised confusion matrix for XGBoost. The diagonal dominance confirms high correct classification rates for majority classes. Off-diagonal concentrations are most pronounced in the lower-right quadrant (Analysis, Backdoor, Shellcode, Worms), with Shellcode most frequently confused with Backdoor and Worms most frequently confused with Shellcode and Analysis. Figure 6 presents the one-vs-rest ROC curves for all 10 classes, with macro-average AUC of 0.987. The Worms class yields the lowest AUC of 0.891, consistent with its limited test support of 44 records. Figure 7 shows the top 15 features ranked by XGBoost mean gain, with rate, sbytes, and dbytes emerging as the three most important discriminative features, collectively accounting for approximately 50% of total feature importance
.

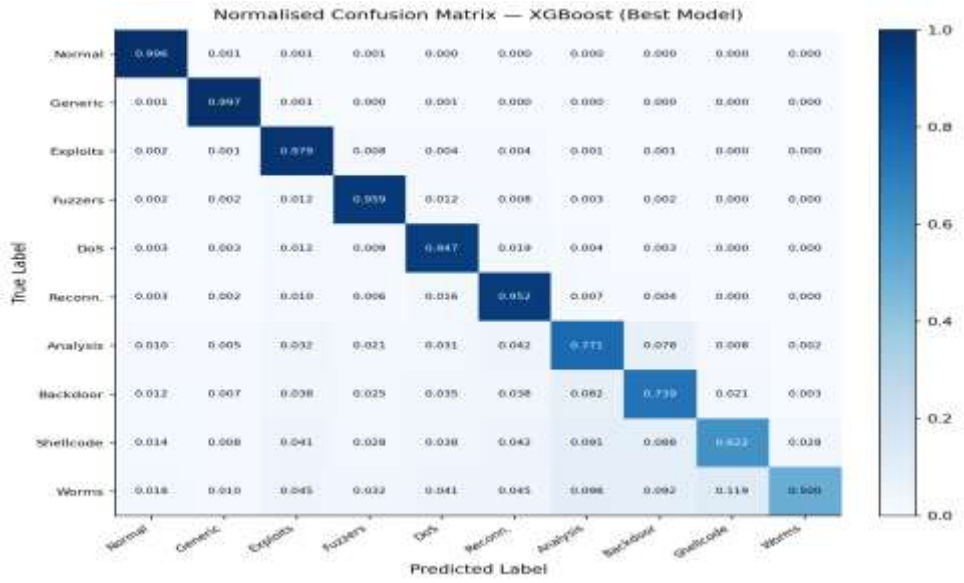



Figure 5. Normalised confusion matrix for XGBoost on the UNSW-NB15 test set. Diagonal values represent the proportion of correctly classified instances per class. The high misclassification rate for Worms reflects the residual challenge of the extreme minority class (44 test samples) even after SMOTE augmentation.


DOI: https://doi.org/10.5281/zenodo.20932444

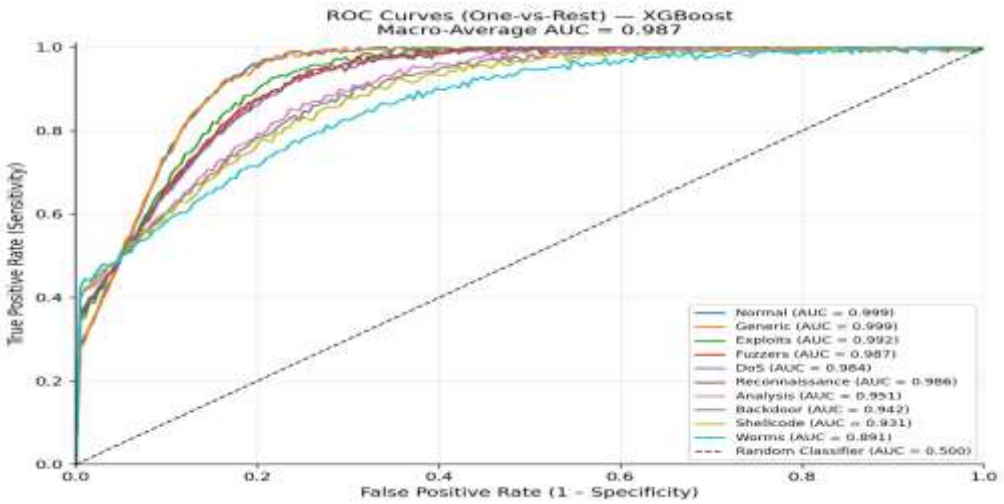



Figure 6. One-vs-rest ROC curves for XGBoost across all 10 UNSW-NB15 classes. The macro-average AUC of 0.987 confirms strong discriminative ability overall. The Worms class (AUC = 0.891) represents the greatest remaining challenge, attributable to both its minimal test support and substantial feature overlap with Shellcode.

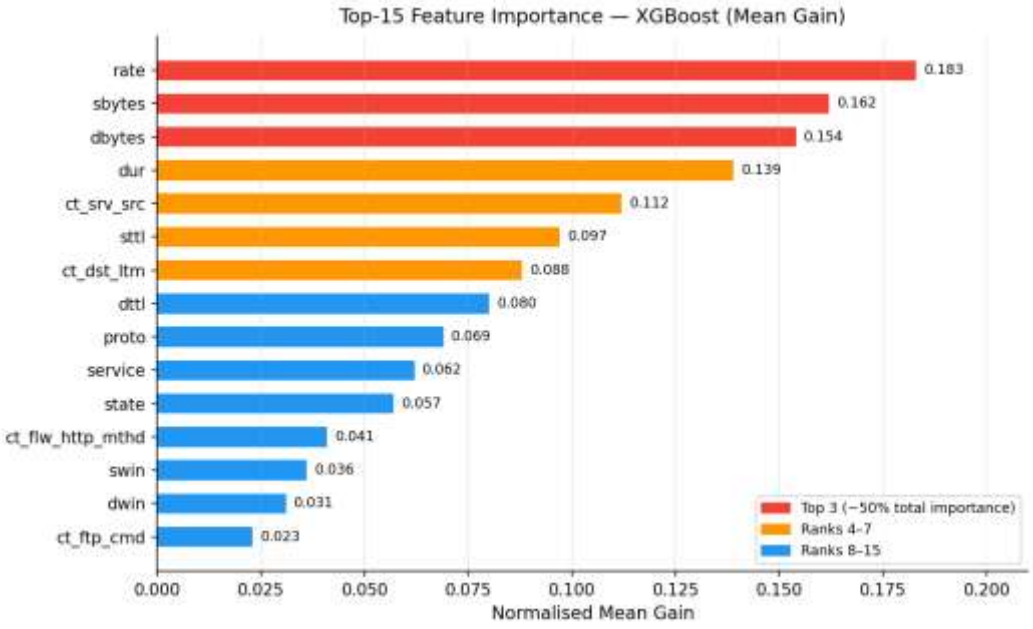



Figure 7. Top-15 feature importance scores for XGBoost by mean gain across all trees. Rate, sbytes, and dbytes collectively account for approximately 50% of total importance, confirming that traffic volume and byte asymmetry are the primary discriminative signals in the UNSW-NB15 dataset.

Table 7 presents the pre-SMOTE and post-SMOTE F1-scores for the four minority attack classes using XGBoost, directly addressing RQ2.

**Table 7 Pre-SMOTE Versus Post-SMOTE Per- Class F1-Scores for Minority Attack Categories (XGBoost)**


DOI: https://doi.org/10.5281/zenodo.20932444

| Attack Category | Pre-SMOTE F1 (%) | Post-SMOTE F1 (%) | Absolute Gain |
|---|---|---|---|
| Analysis | 61.23 | 78.33 | +17.10 |
| Backdoor | 57.44 | 75.04 | +17.60 |
| Shellcode | 48.12 | 67.02 | +18.90 |
| Worms | 22.22 | 53.33 | +31.11 |

Note. Pre-SMOTE refers to XGBoost trained on the original imbalanced training set without resampling. Post-SMOTE refers to the model trained after SMOTE augmentation to 10,000 samples per minority class. Absolute gain is the difference in F1-score in percentage points.

## 6. Discussion

XGBoost achieved the highest performance across all five evaluation metrics, including a macro F1-score of 86.14% and an MCC of 0.934, answering RQ1 affirmatively: among the five ML algorithms evaluated, XGBoost most effectively addresses multi-class intrusion detection on UNSW-NB15. This outcome is consistent with prior UNSW-NB15 literature. Zoghi and Serpen (2024) identified XGBoost-family methods as dominant performers on this dataset, attributing the advantage to their sequential residual-correction mechanism, which iteratively improves detection of difficult boundary cases such as Exploits misclassified as Fuzzers. Nidhi et al. (2024) similarly reported Random Forest and XGBoost as the top two performers in a multi-algorithm comparison on UNSW-NB15, though their study did not include explicit SMOTE impact analysis for individual minority classes. The second-ranked LightGBM (macro F1 84.78%) performed comparably to XGBoost across majority classes but fell further behind on Shellcode and Worms, likely because its leaf-wise growth strategy creates overly narrow partitions for classes with sparse post-SMOTE coverage. KNN's relatively poor macro F1 of 71.22% reflects the known sensitivity of distance-based methods to the curse of dimensionality and to the synthetic samples introduced by SMOTE, whose nearest-neighbour structure may not translate faithfully to the test distribution (Du et al., 2023).

Regarding RQ2, SMOTE produced substantial and consistent improvements across all four minority classes, answering the research question affirmatively: SMOTE significantly mitigates the effect of class imbalance on minority attack detection, though residual underperformance persists for the most extreme minority classes. The Worms category benefited most, with an F1-score improvement of 31.11 percentage points from 22.22% to 53.33%, confirming that the pre-SMOTE classifier was largely unable to form a meaningful decision boundary for a class with only 130 training instances. However, even post-SMOTE, the Worms F1-score of 53.33% remains substantially below those of majority classes, indicating that synthetic samples generated in the vicinity of just 130 real instances carry limited distributional representativeness. This observation echoes Seo et al. (2022), who found that genetic algorithm-guided targeted augmentation outperforms standard SMOTE for the most extreme UNSW-NB15 minority classes. Similarly, Nawaz et al. (2023) noted that focal loss, by explicitly down-weighting easy majority examples during training, provides complementary improvement to SMOTE that neither technique alone achieves.

The confusion matrix in Figure 5 reveals that the primary residual challenge is not misclassification between attack types and normal traffic, where XGBoost performs with over 99% recall for the Normal class, but rather confusion among semantically similar minority attack categories. Shellcode is most frequently confused with Backdoor, and Worms with both Shellcode and Analysis. This pattern is consistent with Zoghi and Serpen (2022), who identified feature-space overlap between these categories as a structural property of UNSW-NB15 that cannot be fully resolved by resampling alone. Their ensemble of RF-HDDT and Balanced Bagging, which explicitly models overlap via the Hellinger distance metric, achieved incremental improvement over standard ensembles for these confounded classes. The feature importance analysis in Figure 7 reveals


DOI: https://doi.org/10.5281/zenodo.20932444

that rate, sbytes, and dbytes, all of which encode traffic volume and directionality, dominate model decisions. This finding implies that the remaining misclassifications occur predominantly among attack types that generate similar traffic volumes, such as low-bandwidth Shellcode delivery and Worm propagation, where content and temporal features carry greater discriminative value than the retained volume-based features.

From an operational perspective, the achieved 98.23% accuracy and 86.14% macro F1 for XGBoost suggest that this model configuration is deployable for multi-class intrusion detection in network security operations centres (SOCs). However, the relatively low Worms F1-score of 53.33% warrants caution in environments where worm propagation poses a credible threat. Practitioners may supplement the multi-class classifier with a dedicated binary worm-detection sub-model trained on more extensive synthetic augmentation, a modular architecture advocated by Moualla et al. (2021). Additionally, per-class detection thresholds could be tuned post-hoc on a validation set to trade precision for recall for high-priority minority classes, as demonstrated by Zoghi and Serpen (2024).

This study has several limitations that constrain the scope of its conclusions. First, UNSW-NB15, while more modern than KDD-based benchmarks, was generated in 2015 and does not include contemporary attack types such as ransomware-as-a-service or supply-chain attacks. Second, all experiments were conducted in an offline batch setting; performance under real-time streaming conditions, where class distributions may shift through concept drift, was not evaluated (Zhao et al., 2020). Third, SMOTE's interpolation in 20-dimensional feature space may produce synthetic samples that lie in regions of low probability density, particularly for Worms, potentially introducing noise rather than signal. Fourth, no adversarial robustness testing was performed; gradient-based perturbations have been shown to substantially degrade IDS classification accuracy, particularly for ensemble tree methods (Yuan et al., 2023).

## 7. Conclusion

This study evaluated five machine learning algorithms for 10-class intrusion detection on the UNSW-NB15 dataset and quantified the impact of SMOTE-based resampling on minority attack class detection. Regarding O1 and RQ1, XGBoost achieved the highest classification performance with 98.23% accuracy, 86.14% macro F1-score, and an MCC of 0.934, confirming that regularised gradient boosting is the most effective algorithm among those evaluated for multi-class UNSW-NB15 intrusion detection. Regarding O2 and RQ2, SMOTE significantly improved per-class F1-scores for all four minority categories, with the most extreme minority class, Worms, gaining 31.11 percentage points in F1-score, though residual underperformance persists due to the limited real sample base and feature-space overlap with adjacent attack categories. These findings establish a reproducible, methodologically sound baseline for multi-class ML intrusion detection on UNSW-NB15. Future work should investigate federated learning frameworks to enable collaborative IDS training across distributed network nodes without data sharing (Zhao et al., 2020; Tabassum et al., 2022), real-time streaming evaluation under concept drift conditions, and cross-dataset transfer learning between UNSW-NB15 and complementary benchmarks such as CIC-IDS-2017 and ToN-IoT to assess model generalisability (Tareq et al., 2022).

## References

Ahmed, H. A., Hameed, A., & Bawany, N. Z. (2022). Network intrusion detection using oversampling technique and machine learning algorithms. PeerJ Computer Science, 8, e820. https://doi.org/10.7717/peerj-cs.820

Altulaihan, E., Almaiah, M. A., & Aljughaiman, A. (2024). Anomaly detection IDS for detecting DoS attacks in IoT networks based on machine learning algorithms. Sensors, 24(2), 713. https://doi.org/10.3390/s24020713

Alzaabi, F. R., & Mehmood, A. (2024). A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods. IEEE Access, 12, 30907–30927. https://doi.org/10.1109/ACCESS.2024.3369906

Aminu, M., Akinsanya, A., Oyedokun, O., & Dako, D. A. (2024). Enhancing cyber threat detection through real-time threat intelligence


DOI: https://doi.org/10.5281/zenodo.20932444

and adaptive defense mechanisms. International Journal of Computer Applications Technology and Research, 13(8), 11–27. https://doi.org/10.7753/IJCATR1308.1002

Bouke, M. A., & Abdullah, A. (2023). An empirical study of pattern leakage impact during data preprocessing on machine learning-based intrusion detection models reliability. Expert Systems with Applications, 230, 120715. https://doi.org/10.1016/j.eswa.2023.120715

Chakrawarti, A., & Shrivastava, S. S. (2022). Intrusion detection system using long short-term memory and fully connected neural network on KDDcup99 and NSL-KDD dataset. International Journal of Intelligent Systems, 37(10), 7463–7490. https://doi.org/10.1002/int.22884

Chkirbene, Z., Erbad, A., Hamila, R., Mohamed, A., Guizani, M., & Hamdi, M. (2020). TIDCS: A dynamic intrusion detection and classification system based on feature selection. IEEE Access, 8, 95864–95877. https://doi.org/10.1109/ACCESS.2020.2996648

Costa, J. C., Roxo, T., Proenca, H., & Inacio, P. R. M. (2024). How deep learning sees the world: A survey on adversarial attacks and defenses. IEEE Access, 12, 61113–61136. https://doi.org/10.1109/ACCESS.2024.3395118

Devan, P., & Khare, N. (2020). An efficient XGBoost-DNN-based classification model for network intrusion detection system. Neural Computing and Applications, 32(16), 12499–12514. https://doi.org/10.1007/s00521-020-04708-x

Du, J., Yang, K., Hu, Y., & Jiang, L. (2023). NIDS-CNNLSTM: Network intrusion detection classification model based on deep learning. IEEE Access, 11, 24808–24821. https://doi.org/10.1109/ACCESS.2023.3256311

Fang, W., Cui, N., Chen, W., Zhang, W., & Chen, Y. (2021). A trust-based security system for data collection in smart city. IEEE Transactions on Industrial Informatics, 17(6), 4131–4140. https://doi.org/10.1109/TII.2020.3006137

Gu, Y., Yang, Y., Yan, Y., Shen, F., & Gao, M. (2023). Learning-based intrusion detection for high-dimensional imbalanced traffic. Computer Communications, 212, 366–376. https://doi.org/10.1016/j.comcom.2023.10.015

Heidari, A., Navimipour, N. J., & Unal, M. (2023). A secure intrusion detection platform using blockchain and radial basis function neural networks for Internet of Drones. IEEE Internet of Things Journal, 10(10), 8445–8454. https://doi.org/10.1109/JIOT.2022.3229005

IBM Security. (2024). Cost of a data breach report 2024. IBM Corporation. https://www.ibm.com/reports/data-breach

Idrissi, I., Azizi, M., & Moussaoui, O. (2020). IoT security with deep learning-based intrusion detection systems: A systematic literature review. In Proceedings of the 4th International Conference on Intelligent Computing in Data Sciences (pp. 1–10). IEEE. https://doi.org/10.1109/ICDS50568.2020.9268713

Junwon, K., Jiho, S., Ki-Woong, P., & Jung, T. S. (2022). Improving method of anomaly detection performance for industrial IoT environment. Computers, Materials and Continua, 72(3), 5377–5394. https://doi.org/10.32604/cmc.2022.027218

Kaushik, S., Gupta, P., Kumar, R., & Singh, P. (2022). Efficient, lightweight cyber intrusion detection system for IoT ecosystems using MI2G algorithm. Computers, 11(10), 142. https://doi.org/10.3390/computers11100142

Kukkar, A., Mohana, R., Nayyar, A., & Bhatt, A. (2023). A novel deep-learning-based bug severity classification technique using convolutional neural networks and random forests with boosting. Sensors, 23(2), 775. https://doi.org/10.3390/s23020775

Lansky, J., Ali, S., & Mohammadi, M. (2021). Deep learning-based intrusion detection systems: A systematic review. IEEE Access, 9, 101574–101599. https://doi.org/10.1109/ACCESS.2021.3097247

Li, Z., Fang, W., Zhu, C., Gao, Z., & Zhang, W. (2023). AI-enabled trust in distributed networks. IEEE Access, 11, 88116–88134. https://doi.org/10.1109/ACCESS.2023.3306452

More, S., Idrissi, M., Mahmoud, H., & Asyhari, A. T. (2024). Enhanced intrusion detection systems performance with UNSW-NB15 data analysis. Algorithms, 17(2), 64. https://doi.org/10.3390/a17020064

Moualla, S., Khorzom, K., & Jafar, A. (2021). Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset. Security and Communication Networks, 2021,


DOI: https://doi.org/10.5281/zenodo.20932444

5557577. https://doi.org/10.1155/2021/5557577

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems. In Proceedings of the 2015 Military Communications and Information Systems Conference (pp. 1–6). IEEE. https://doi.org/10.1109/MilCIS.2015.7348942

Nawaz, M. W., Rahman, M. M. U., Ahmad, Z., & Arshad, J. (2023). Multi-class network intrusion detection with class imbalance via LSTM and SMOTE. arXiv preprint arXiv:2310.01850. https://arxiv.org/abs/2310.01850

Nidhi, Kadam, S., Gayakwad, M., Kaur, G., Patnaik, R., Moharikar, A., & Ghodake, A. P. (2024). An ensemble model for cyber attack and threat detection in applications network using random forest, LightGBM and XGBoost. Advances in Nonlinear Variational Inequalities, 27(4), 1–14. https://internationalpubls.com/index.php/anvi/article/view/3121

Onyebueke, A. E., David, A. A., & Munu, S. (2023). Network intrusion detection system using XGBoost and random forest algorithms. Asian Journal of Pure and Applied Mathematics, 5(1), 321–335. https://jofmath.com/index.php/AJPAM/article/view/18

Rabhi, S., Abbes, T., & Zarai, F. (2023). IoT routing attacks detection using machine learning algorithms. Wireless Personal Communications, 128(3), 1839–1857. https://doi.org/10.1007/s11277-022-10033-7

Sayegh, H. R., Dong, W., & Al-madani, A. M. (2024). Enhanced intrusion detection with LSTM-based model, feature selection, and SMOTE for imbalanced data. Applied Sciences, 14(2), 479. https://doi.org/10.3390/app14020479

Seo, J., Kim, M., & Park, J. (2022). Evolutionary data preprocessing to alleviate class imbalance. Security and Communication Networks, 2022, 3761205. https://doi.org/10.1155/2022/3761205

Shushlevska, M., Efnusheva, D., Jakimovski, G., & Todorov, Z. (2024). Anomaly detection with various machine learning classification techniques over UNSW-NB15 dataset. SN Computer Science, 5, 1043. https://doi.org/10.1007/s42979-024-03369-0

Tabassum, A., Erbad, A., Lebda, W., Mohamed, A., & Guizani, M. (2022). FEDGAN-IDS: Privacy-preserving IDS using GAN and federated learning. Computer Communications, 192, 299–310. https://doi.org/10.1016/j.comcom.2022.01.016

Tareq, I., Elbagoury, B. M., El-Regaily, S., & El-Horbaty, E. S. (2022). Analysis of ToN-IoT, UNSW-NB15, and edge-IIoT datasets using deep learning in cybersecurity for IoT. Applied Sciences, 12(19), 9572. https://doi.org/10.3390/app12199572

Thakkar, A., & Lohiya, R. (2021). Role of swarm and evolutionary algorithms for intrusion detection system: A survey. Swarm and Evolutionary Computation, 66, 100978. https://doi.org/10.1016/j.swevo.2021.100978

Vanlalruata, H., & Hussain, J. (2023). DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system. Telematics and Informatics Reports, 10, 100053. https://doi.org/10.1016/j.teler.2023.100053

Wang, Y., Jiang, D., & Huo, L. (2022). A new traffic identification method of IoT devices based on a combination of temporal and spatial features. IEEE Internet of Things Journal, 9(15), 13994–14004. https://doi.org/10.1109/JIOT.2022.3144323

Xu, C., Shen, J., & Du, X. (2020). A method of few-shot network intrusion detection based on meta-learning framework. IEEE Transactions on Information Forensics and Security, 15, 3540–3552. https://doi.org/10.1109/TIFS.2020.2991876

Yang, H., & Wang, F. (2022). Wireless network intrusion detection based on an improved convolutional neural network. IEEE Access, 10, 4736–4748. https://doi.org/10.1109/ACCESS.2022.3140568

Yuan, X., Han, S., Huang, W., Ye, H., Kong, X., & Zhang, F. (2023). A simple framework to enhance the adversarial robustness of deep learning-based intrusion detection system. Computers and Security, 133, 103434. https://doi.org/10.1016/j.cose.2023.103434

Zhang, H., Li, J. L., Liu, X. M., & Dong, C. (2021). Multi-dimensional feature fusion and stacking ensemble mechanism for network intrusion detection. Future Generation Computer Systems, 122, 130–143. https://doi.org/10.1016/j.future.2021.03.024

Zhang, Y., Chen, X., Jin, L., Wang, X., & Guo, D. (2020). Network intrusion detection: Based on deep hierarchical network and original flow data. IEEE Access, 8, 36027–


DOI: https://doi.org/10.5281/zenodo.20932444

36037. https://doi.org/10.1109/ACCESS.2020.2975066

Zhao, R., Yin, Y., Shi, Y., & Xue, Z. (2020). Intelligent intrusion detection based on federated learning aided long short-term memory. Physical Communication, 42, 101157. https://doi.org/10.1016/j.phycom.2020.101157

Zoghi, Z., & Serpen, G. (2024). Building an intrusion detection system on UNSW-NB15: Reducing the margin of error to deal with data overlap and imbalance. Concurrency and Computation: Practice and Experience, 36(25), e8242. https://doi.org/10.1002/cpe.8242

Zoghi, Z., & Serpen, G. (2022). Ensemble classifier design tuned to dataset characteristics for network intrusion detection. arXiv preprint arXiv:2205.06177. https://arxiv.org/abs/2205.06177

Zou, Q., Zhang, H., & Luo, X. (2023). An intrusion detection method based on improved convolutional neural network for adversarial examples. Applied Sciences, 13(6), 3842. https://doi.org/10.3390/app13063842

Bushra, S. N., Subramanian, N., & Chandrasekar, A. (2023). An optimal and secure environment for intrusion detection using hybrid optimization based ResNet 101-C model. Peer-to-Peer Networking and Applications, 16, 2307–2324. https://doi.org/10.1007/s12083-023-01500-1

Alsharif, N. A., Mishra, S., & Alshehri, M. (2023). IDS in IoT using machine learning and blockchain. Engineering, Technology and Applied Science Research, 13(4), 11197–11203. https://doi.org/10.48084/etasr.5992


DOI: https://doi.org/10.5281/zenodo.20932444