CINEFORUM
ISSN: 0009-7039
Vol. 66. No. 2, 2026

Intrusion Detection System for AMI in Smart Grid Using Hybrid Tree-based Classifier

Indranil Das[1], Ranjan Banerjee[2], Debmalya Mukherjee[3], Shankar Prasad Mitra[4], Amartya Ghosh[5], BRATIN DAS[6], ATANU KUMAR DAS[7], Niloy Biswas[8], Sumit Roy[9]

[1]Assistant Professor, Computer Science & Engineering, Brainware University, Orcid Id: 0009-0005-6744-8832, indranil199820das@gmail.com
[2]Assistant Professor, Computer Science & Engineering, Brainware University, Orcid Id:0009-0003-1950-7530, rbkpccst@gmail.com
[3]Assistant Professor, Computational Sciences, Brainware University, Orcid Id: 0009-0006-9946-0964, dbml.mukherjee@gmail.com
[4]Assistant Professor , Computer Science & Engineering, Brainware University, Orcid Id: 0009-0003-5457-5534, spmitra2016@gmail.com
[5]Assistant Professor , Computer Science & Engineering, Brainware University, ORCID Id: 0009-0002-2504-3325, com.amartya@gmail.com
[6]Assistant Professor, Computer Science & Engineering, Adamas University, Orcid Id: 0009-0007-7400-9816, **bratin.das@gmail.com**
[7]Assistant Professor, Computer Science & Engineering, Brainware University, Orcid Id: 0009-0004-6974-1875, atanu.csh@gmail.com
[8]Assistant Professor, Computer Science & Engineering, Brainware University Orcid Id: 0009-0003-5644-0478, niloyb1990@gmail.com
[9]Assistant Professor, Computer Science & Engineering, Brainware University, Orcid Id:0009-0000-6312-3153 sumit.oficial.kol@gmail.com

Abstract

The current research suggests an IDS for AMI in smart grids as an effective approach to the detection of any threats. The approach utilizes a hybrid tree-based classifier along with information gain as a method of feature selection in order to improve the performance of detection in the neighborhood area network (NAN).

For the identification of the features that should be selected from the data, information gain is utilized, as it shows the relation between the features and the attack type. In addition, the classifier is enhanced by the utilization of the stacking algorithm.

The performance of the suggested model was tested using the UNSW-NB15 dataset. It turned out that the suggested approach is more efficient than other machine learning approaches such as Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), and K-Nearest Neighbors (KNN).

Keywords: Intrusion Detection System (IDS), Advanced Metering Infrastructure (AMI), Smart Grids , Neighborhood Area Network (NAN), Feature Selection , Information Gain (IG), Hybrid Tree-Based Classifier, Stacking Algorithm, UNSW-NB15 Dataset

Introduction

Availability of electricity is a basic necessity in today's world, used in households, businesses, industries, and transportation. The growing needs of power have brought about the need for more efficient and reliable systems of production, transmission, distribution, and consumption of energy. This has led to the development of smart grids that incorporate the Advanced Metering Infrastructure (AMI).

The smart grid is an advanced electrical system that incorporates the traditional electrical systems with communication, control, and information technology systems in order to facilitate communication and exchange of information in real time.

The Advanced Metering Infrastructure (AMI) is a major part of the smart grid and comprises the smart meter, communication system, and data management system. The smart meters continually record the use of electricity and relay the information in real time. The benefits of AMI include consumer awareness, energy efficiency, and adoption of time-based pricing of energy.

Need for Intrusion Detection Systems in AMI

As AMI plays a very important role in the smart grid system, it becomes prone to different types of security risks. IDS is a very important tool to guarantee that AMI remains safe from any kinds of security risks. IDS is used for detecting any kind of intrusion like hacking and cyber attacks like phishing.

Moreover, AMI has access to the personal information of consumers, which makes it necessary to protect that data.

OVERVIEW OF THE SMART GRID

A smart grid is an advanced energy transmission system incorporating communication technology, digital architecture, and intelligent controls. This differs from conventional grids since it allows the bidirectional flow of power and information between utilities and consumers.

This comprises smart meters, sensors, automated control systems, and analytical tools. Smart meters enable real-time tracking of energy consumption and allow two-way communication, while sensors continually monitor the status of the grid.

OVERVIEW OF ADVANCED METERING INFRASTRUCTURE (AMI)

Advanced Metering Infrastructure (AMI) is a system that integrates smart meters, communication networks, and data management systems to enable two-way communication between utilities and consumers. It provides real-time electricity consumption data, improving monitoring and management of the smart grid.

A. Components of AMI

1. **Smart Meters:** Devices installed at consumer premises to measure and transmit electricity usage data.
2. **Communication Network:** Enables bidirectional data exchange between smart meters and utility providers.
3. **Data Management System:** Processes, stores, and analyzes data for efficient operation and billing.
4. **Head-End System:** Acts as the central hub that collects, validates, and forwards data to the management system.

Hierarchical Architecture of AMI Communication Networks

- **The Advanced Metering Infrastructure (AMI)** depends on the hierarchical structure for bidirectional communication between customer edge devices and the central utility systems [1]. The hierarchical structure has been categorized into three operational domains:
- **Home Area Network (HAN):** Operational at the localized edge of the consumer end and responsible for communicating between intelligent in-home appliances, devices, and the customer side of the smart meter [1, 2]. It uses low-power technologies such as IEEE 802.15.4 (Zigbee) or wireline power line communication (PLC).
- **Neighborhood Area Network (NAN):** Acts as an intermediary aggregation level for connecting a number of smart meters in a local geographic area and transferring telemetry data to a centralized Data Concentration Unit (DCU) [1, 2]. Some of the implementation choices may be wireless mesh networking, WiMAX, and cellular based networks (LTE-M, NB-IoT).
- **Wide Area Network (WAN):** It acts as high-speed and long-distance communication backhaul between distributed DCUs and core utility infrastructure, which includes the Head-End System (HES) and Meter Data Management System (MDMS) [1, 2]. As it has very high bandwidth needs, it is implemented using reliable fiber optics, DSL, or a long-range core cellular network.

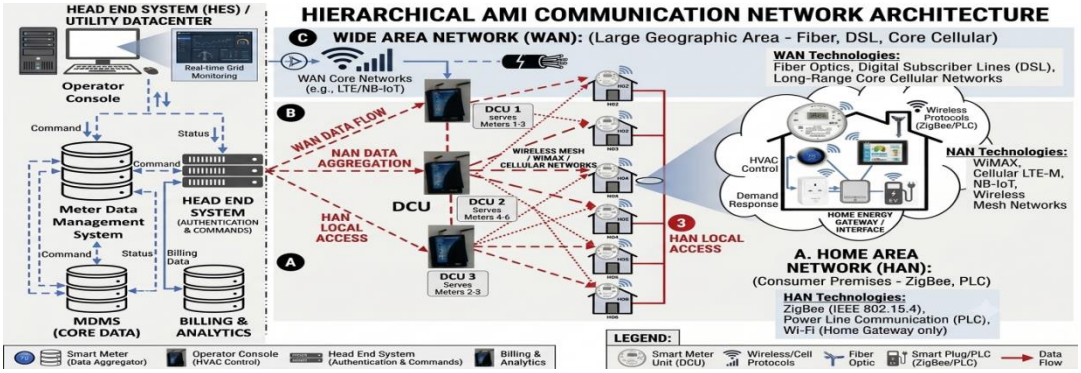


Fig. 1 AMI communication network

Related Work

Optimization of Intrusion Detection Systems (IDS) in Advanced Metering Infrastructure (AMI) continues to be an essential research field in smart grid cybersecurity due to the heterogeneous communication layers and limited resources of devices used. Vijayanand et al. [3] suggested MI-based feature selection together with a multi-class SVM ensemble for anomaly detection in the NAN layer. The authors managed to reduce features to five important parameters and got 90.0% accuracy, which is significantly better than that of a non-optimized algorithm (76.2%), though SVM requires time for calculations.

Multi-layer security challenges have been addressed in a distributed IDS architecture that operates in HAN, NAN, and WAN layers [4]. With KDD99 dataset used, SVM showed a detection rate of 97.5%, but FAR was rather low. As for DT, its performance is comparable. However, high computation cost limits the practical implementation of the solution on resource-limited devices.

Also, meta-heuristics like PSO have been used in feature selection [5]. Upon combination of NN, KNN, RF, and DT, the model scored an accuracy of 99.65% using NSL-KDD. Although it scored very highly, PSO's iterations make it not dynamic to threats that happen in real time.

Models using gradient boosting methods like LightGBM have proved to be very accurate on modern datasets such as UNSW-NB15 [6]. Using the reduction and encoding method, the model achieved an accuracy of 95.19% and efficient runtime performances.

Ensembles that use stacking improve detection of threats in complex situations [7]. An example is the multi-layer ensembling approach that uses RF, ANN, and DT achieving a high accuracy on UNSW-NB15 and CICIDS2017 datasets. Nevertheless, the ANN part makes computation difficult and uninterpretable.

The application of bagging techniques has also been used for gaining stability, which gives moderate accuracy but lower FARs at the expense of lowering transparency [6]. Lastly, comparative evaluations using different classifiers have shown the consistent performance of RF under simulation and practical scenarios, although with degradation in dynamic traffic situations [8].

Detailed Comparative Analysis of Literature Intrusion Detection Implementations

Ref.	Methodology / Algorithmic Architecture	Target Domain	Evaluation Dataset	Key Performance Metrics	Identified Limitations / Disadvantages

[3]	Multi-Class SVM with Mutual Information (MI) feature selection	Neighborhood Area Network (NAN)	ADFA-LD	Accuracy: 90.0% DR (Normal): 93.4% DR (Attack): 89.2%	High matrix inversion complexity for large-scale data
[4]	Tiered SVM, KNN, and Decision Tree across network layers	HAN, NAN, WAN	KDD99	SVM: Acc 97.5%, FAR 1.3% KNN: Acc 88.9%, FAR 85.2% DT: Acc 95.5%, FAR 3.4%	Poor scalability and memory bottlenecks in SVM layer
[5]	PSO-optimized NN, KNN, RF, DT classifiers	Enterprise Substation Layer	NSL-KDD, KDD99	NN (NSL-KDD): Acc 99.65%, FAR 0.08% NN (KDD99): Acc 99.20%, DR 99.70%	High computational cost due to iterative PSO convergence
[6]	LightGBM with RF-based feature importance ranking	AMI Core Infrastructure	UNSW-NB15	Accuracy: 95.19% Precision: 96.84% Recall: 95.58% FAR: 4.81%	Sensitive to hyperparameter tuning under varying conditions
[7]	Stacking (RF + ANN) with DT meta-learner and IG feature selection	Smart City IoT Edge Systems	UNSW-NB15, CICIDS2017	UNSW-NB15: Acc 96.83%, Prec 97.0% CICIDS2017: Acc 99.9%, Prec 99.9%	High computational cost and ANN-induced opacity

[8]	Bagging ensemble with Information Gain feature reduction	Cloud-Physical Interconnects	NSL-KDD	Accuracy: 84.93% FAR: 2.45%	Limited interpretability due to bootstrap aggregation
[9]	RF, KNN, BernoulliNB, Linear SVM comparison	Enterprise Network Backhauls	UNSW-NB15, Real Traffic	RF: Acc 97.02%, F1 94.19% (dataset) RF: Acc 95.49%, F1 41.27% (real traffic)	Performance drop under real-world noisy traffic

Research Syntheses and Core Findings

Based on the empirical evidence from contemporary literature, the following structural conclusions guide the selection of algorithmic components for a smart grid intrusion detection system (IDS):

Ensemble Architecture Selection: Stacking models using computationally heavy classifiers (e.g., deep artificial neural networks) introduces high latency and reduced interpretability at the network edge [7]. In contrast, tree-based stacking architectures provide faster training and inference with better interpretability, making them suitable for low-latency environments such as Data Concentrator Units (DCUs).

Feature Selection Synergy: Information Gain (IG) effectively identifies high-relevance features from volatile smart grid data [3], [7], [8]. However, combining IG with non-tree classifiers may reduce accuracy, as IG aligns more naturally with tree-based models by optimizing split entropy for better decision boundaries.

Dataset-Specific Performance: On highly imbalanced datasets like UNSW-NB15, Random Forest (RF) and LightGBM show strong performance [6], [9]. RF offers high stability and strong minority-class detection through decorrelated bootstrap trees, making it effective for detecting rare attacks such as fuzzers and exploits. LightGBM enables fast real-time inference using leaf-wise growth and Gradient-based One-Side Sampling (GOSS), reducing memory usage and computational delay in large-scale deployments.

Algorithmic Synergy and Optimization Targets

Structural Component	Operational Advantage	Smart Grid Constraint Addressed	Core References
Tree-Based Stacking	Multi-tier validation logic without deep learning matrix inflation.	Minimizes data processing latency on edge hardware.	[7]
Information Gain (IG)	Strips noisy parameters by measuring dataset split entropy.	Matches the underlying node mechanics of tree algorithms.	[3], [7], [8]
Random Forest (RF)	Robust bootstrap aggregation balances minority-class weight.	Maintains high target recall under skewed attack balances.	[9]
LightGBM Engine	Leaf-wise splitting with structural memory bundling (GOSS).	Enables real-time, high-volume data stream parsing.	[4]

Problem Statement

Developing a resilient security framework for Advanced Metering Infrastructure (AMI) is challenging due to high traffic volume and complex attack variations. The core technical problems addressed by this research are the following:

Inadequacy of Conventional Binary Classification Paradigms

Relying exclusively on binary models (classifying traffic simply as "benign" or "anomalous") exposes smart grid systems to structural risks:

- **Taxonomy Limitations:** AMI networks are vulnerable to diverse threats (e.g., DoS, DDoS, data injection). A simple binary model cannot identify specific attack variants [10], which prevents downstream systems from executing targeted mitigation strategies.
- **Lack of Severity Scaling:** Critical infrastructures require threat prioritization. Binary frameworks fail to provide fine-grained detection needed to differentiate low-risk operational anomalies from high-risk security breaches [11].

- **Multi-Stage Constraints:** Modern intrusion detection architectures utilize multi-stage pipelines (preprocessing, feature extraction, and analysis). A standalone binary classifier cannot adequately support these sophisticated backend multi-class layers [12].

Structural Vulnerabilities of Single-Classifier Typologies

Relying on a single machine learning model creates analytical bottlenecks when parsing complex, high-dimensional telemetry streams:

- **Attack Pattern Diversity:** Multiclass anomalies exhibit distinct signatures across HAN, NAN, and WAN boundaries. A single classifier cannot simultaneously adapt its rigid decision boundary to all attack vectors [13].
- **Class Imbalance Biasing:** Normal telemetry vastly outnumbers malicious records. A standalone classifier naturally biases its boundaries toward the majority class, leading to high false negative rates for rare minority exploits [14].
- **Absence of Algorithmic Complementarity:** Individual algorithms have unique performance limits. Deploying an ensemble configuration balances individual weaknesses by combining complementary models to capture complex traffic features [15].

Proposed Methodology and Architecture

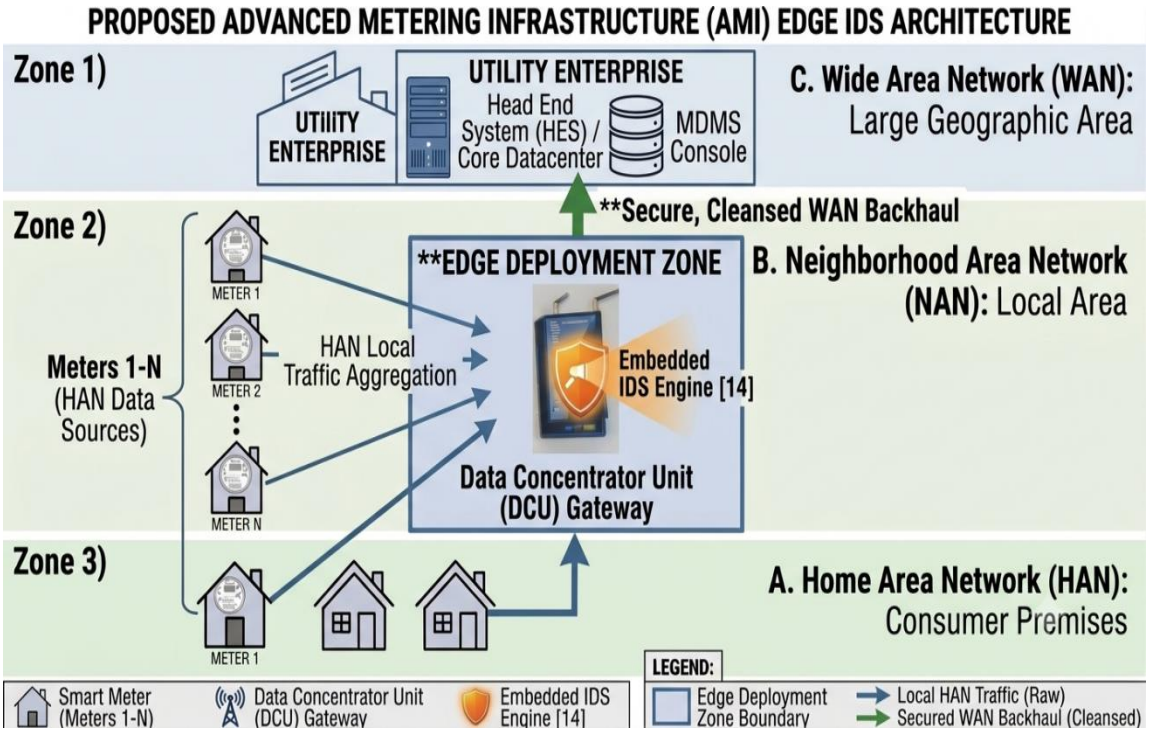
The design of the security framework consists of two main operational stages including strategic system deployment and multistage IDS architecture. The decoupling of physical network topology from the machine learning pipeline allows achieving good visibility in the edge layer and efficient processing.

- **System Design (Strategic Deployment):**
  This stage determines the most effective deployment of IDS nodes in the hierarchical structure of Advanced Metering Infrastructure (AMI) that will help to ensure that malicious activities will be detected at strategic data aggregation points.
- **IDS Architecture (Algorithmic Pipeline):**
  This stage describes the software workflow and modules of the IDS system. It involves all stages of work starting from network packet acquisition and feature selection up to multi-classification using ensemble learning techniques.

Proposed System Design and Deployment Topology

The IDS system is designed to increase visibility of threats while minimizing network latency through its installation at the intermediate edge layer of the NAN architecture in the AMI. The ML component is executed directly on the DCU nodes.

The physical data routing topology and architectural boundaries of this edge deployment are structured as follows:



The DCU acts as the main communication bridge between downstream smart meters in the Home Area Network (HAN) and the central utility server infrastructure. Placing the processing engine at this hub offers advantages over centralized architectures.

- **Cross-Layer Data Inspection:** At the aggregation point, the engine can monitor multiple network layers simultaneously, detecting anomalies as data moves from edge devices to the core system.
- **Operational Cost-Efficiency:** Local processing at the DCU reduces the need to transmit large volumes of raw network data to central servers, decreasing backhaul load and improving the speed of multi-class attack detection [16].
- **Adaptive Security Optimization:** The effectiveness of this distributed setup depends on the computational power of the classifier, network boundaries, and the specific cyber-physical threats targeted within the infrastructure.

Algorithmic IDS Architecture

The internal architecture of the suggested intrusion detection engine is designed in a linear manner through a pipeline-like multi-stage data processing procedure. The architecture links the raw network data inputs with high-resolution threat notifications using four interconnected modules:

- **Stage 1:** Network Dataset Ingestion This module processes the raw packets at the Data Concentrator Unit (DCU) level. In order to validate the detection algorithms, high-resolution benchmark datasets such as the UNSW-NB15 dataset will be used for training and validation purposes.

- **Stage 2:** Data Preprocessing In this stage, the raw data collected from different networks is transformed into mathematical tensors. It includes extensive cleaning of records, one-hot encoding of non-numerical protocols, and standard distribution of numeric vectors.
- **Stage 3:** Information Gain Feature Selection In order to meet the demanding real-time processing requirements at the edge level, this stage uses an information-theoretic filter. Using the concept of data set entropy, this stage is able to select only those features that are most relevant and eliminate redundant features from consideration.
- **Stage 4:** Multi-Layer Stacking Ensemble In order to convert the feature vectors to classification results of attacks, the final prediction module uses the multi-layer stacking approach. This process involves the use of a large number of tree-based classifiers that identify different classes of multi-class variations in parallel.

Intrusion dataset

The experimental evaluation was conducted using **UNSW-NB15**, a modern, publicly available network security dataset developed at the Cyber Range Lab of UNSW Canberra [17]. Introduced to overcome the limitations of aging benchmarks like KDD 99 and NSL-KDD, UNSW-NB15 reflects contemporary network traffic by mixing real normal activities with low-footprint, sophisticated modern attack vectors [17]. Crucially, the dataset was engineered to eliminate massive data redundancy, preventing machine learning models from becoming biased toward frequently repeated records [17].

Dataset Composition and Splits

The dataset comprises **42 structural features** encompassing categorical, binary, and integer data formats [17]. To facilitate robust model development and validation, the data is partitioned into distinct training and testing subsets:

- **Training Set:** 175,341 records (comprising a standard subset of 117,478 records for specific localized baseline evaluations) [17].
- **Testing Set:** 82,332 records (comprising a standard subset of 57,863 records for specific localized baseline evaluations) [17].

Threat Taxonomy and Attack Categories

Beyond binary classification, the dataset evaluates multi-class granularity across **nine distinct contemporary attack families** [17, 18]. The exact categorical distribution and class counts within the dataset are visualized below in **Fig. 4**.

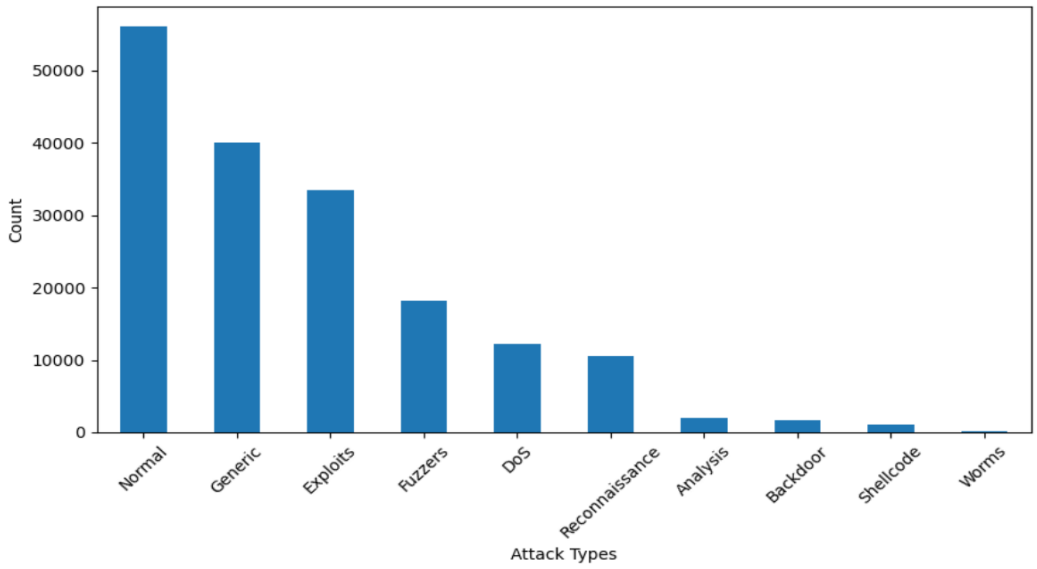


Fig. 4: All attacks listed in the UNSW-NB15 dataset

The characteristic behaviors of these specific attack categories are summarized as follows [16] :

A. Generic Attacks: Techniques that can be applied broadly against block ciphers, independent of specific block or key sizes, without relying on an understanding of the cipher's internal structures [18].

B. Exploits: Target-specific actions where attackers identify software or hardware flaws within systems or IoT devices, weaponizing these vulnerabilities to gain unauthorized access or control [18].

C. Fuzzers: Automated inputs of randomly generated, malformed data injected to disrupt programs or networks, often leading to system crashes or unintended memory behaviors [18].

D. Denial-of-Service (DoS/DDoS): Resource-exhaustion methods engineered to overwhelm target networks, websites, and online services, flooding them with traffic to render them unavailable to legitimate users [18].

E. Reconnaissance: Infrastructure mapping, port scanning, and probing techniques used to gather environmental intelligence and simulate assault scenarios prior to executing an attack [18].

F. Analysis: Web-facing intrusive probing and vulnerability detection, including systematic port scanning, spamming vectors, and malicious HTML file penetrations [16].

G. Backdoors: Stealthily embedded or discovered entry points designed to bypass standard security mechanisms, allowing persistent unauthorized remote access to devices and data [18].

H. Shellcode: Small, precise segments of executable code used as functional payloads to exploit software vulnerabilities and seize control of a target command shell [16].

I. Worms: Autonomous, self-replicating malicious programs that propagate aggressively across active network connections to infect multiple devices and cause widespread damage [18].

Data Preprocessing

The data preprocessing phase transforms raw, heterogeneous network telemetry into structured mathematical representations optimized for machine learning algorithms. This phase primarily details the feature encoding strategy tailored to the categorical attributes of the UNSW-NB15 benchmark dataset [17].

Feature Encoding

The UNSW-NB15 dataset contains a mixture of numeric and categorical variables, featuring three nominal textual attributes: proto (protocol), state (connection state), and attack_cat (threat category) [17, 19]. To prepare these fields for machine learning algorithms, they must be converted into numerical representations [19].

While **one-hot encoding** and **label encoding** are standard approaches [19], this pipeline utilizes label encoding. One-hot encoding maps categories into independent binary columns, which would drastically expand the feature dimensions given the hundreds of unique protocol entries in the proto attribute, creating a sparse, high-overhead matrix [17, 19]. Label encoding eliminates this dimensional expansion by transforming each category into a unique scalar integer within its existing column, maintaining the strict 42-feature footprint for efficient execution [19].

The target categorical dimensions initialized for this integer transformation include [17, 18]:

- **proto:** tcp, udp, unas, arp, ospf, rdp, NetBT, igmp, icmp, rtp, etc.
- **state:** INT, FIN, CON, REQ, RST, ECO, PAR, URN, no.
- **attack_cat:** Normal, Generic, Exploits, Fuzzers, DoS, Reconnaissance, Analysis, Backdoors, Shellcode, Worms.

Feature scaling

Feature scaling is a vital preprocessing step for distance-based machine learning estimators [20]. Because heterogeneous network features manifest across highly disparate numerical boundaries, unscaled attributes with larger ranges can dominate distance metrics, distorting model calculations and introducing algorithmic bias [20]. Normalization standardizes these distinct domains into an equitable, relative scale.

This framework implements **min-max normalization** to linearly transform raw measurements into a uniform, bounded interval between 0 and 1 via the following formulation:

$$F = (F - F_{min}) / (F_{max} - F_{min})$$

where F_{min} and F_{max} show the min and max values of feature F.

Feature Selection

Feature selection is an important aspect of optimization in machine learning, which influences the performance and efficiency of the intrusion detection system (IDS) [21]. High-dimensional datasets like UNSW-NB15 usually consist of redundant or irrelevant features in raw network data, making the computational process expensive and causing overfitting [21]. Feature selection involves determining feature importance in relation to the target and simplifying the model.

The Information Gain Ratio technique will be employed in this experiment. It is a modified version of the Information Gain algorithm proposed by Quinlan [20]. Where Information Gain quantifies the amount of information gained by partitioning data based on a variable, the Information Gain Ratio algorithm compensates for this drawback by normalizing Information Gain through the intrinsic entropy of the variable [22]:

$$Gain\ Ratio(F) = Information\ Gain(F) \div Intrinsic\ Info(F)$$

This resulted in a reduction of the feature space by 50%, from 42 to 21. Features whose score is ≥ 0.4 were selected for further analysis, while others were discarded. The ranking of the selected features along with the scores is shown in Table 2 below.

Table 2: Information Gain Ratio Scores for the Selected 21 Features on the UNSW-NB15 Dataset

No.	Feature Name	Gain Ratio Score	No.	Feature Name	Gain Ratio Score
1	sbytes	1.150814	12	ct_srv_src	0.483279
2	smean	0.927231	13	dload	0.480695
3	sload	0.905526	14	ct_dst_src_ltm	0.480147
4	dbytes	0.635577	15	proto	0.479347
5	rate	0.550283	16	sttl	0.473088
6	dmean	0.547751	17	dinpkt	0.469851
7	dur	0.537349	18	ct_src_dport_ltm	0.468526

8	ct_dst_sport_ltm	0.520676	19	dpkts	0.464674
9	ct_srv_dst	0.506251	20	ct_dst_ltm	0.455093
10	ct_state_ttl	0.494799	21	sinpkt	0.441838
11	dttl	0.490197			

Stacking Ensemble Framework

Stacking ensemble is an approach that involves the use of a diverse set of base classifiers to create a layered model for prediction [23]. Even though it introduces some computational cost, stacking ensembles increases the accuracy of anomaly detection since it lowers the variance and spread of errors [23].

The presented model is based on the integration of three tree-based base classifiers, such as Decision Tree (DT), Random Forest (RF), and LightGBM, benefiting from their respective advantages for IDS:

Increasing the Efficiency of IDS

Tree-based algorithms work quicker compared to distance algorithms like SVM or KNN, which makes tree-based models more appropriate for usage in real-time [24]. The combination of several tree algorithms enables compensation for possible false positives or false negatives.

The proposed model incorporates three base classifiers based on tree ensembles, namely Decision Trees (DT), Random Forest (RF), and LightGBM, leveraging their strengths within an IDS [23, 25]:

Maximizing Classification Accuracy and Compensating Mistakes: Methods based on tree learners perform faster than the distance-based approaches like SVM and KNN and thus can be employed in real-time operation [24]. The combination of several tree learners helps compensate for false positives and false negatives in order to improve multi-class classification results [23].

Better Generalization through Diversity in Models: Diverse methods help to efficiently process heterogeneous data using, for example, bagging and feature subsampling [24]. Diverse methods can detect various patterns in network traffic and therefore can provide more robust protection against unknown or zero-day attacks [23, 25].

Flexibility and Updatable Models: Models based on tree ensembles are flexible to new attack patterns [24]. In the stacking setting, models are flexible and modular, and therefore it is possible to update or replace weaker or outdated classifiers without changing the whole architecture [25].

Overfitting Mitigation and Hyperparameter Optimization

However, while stacking ensembles provide increased accuracy, they could suffer from overfitting [23]. To mitigate this problem, the framework employs 5-fold cross-validation and Bayesian hyperparameter optimization [26, 27], in accordance with the assumption that there is no one best model configuration in light of the No Free Lunch Theorems [28].

The 5-fold cross-validation involves splitting the dataset into five partitions and validating the model over all the folds.

The framework utilizes Bayesian optimization for efficient tuning of the model parameters. It provides more efficient search through the use of probabilistic models, decreases the number of training attempts when compared with grid/random search, and manages the mixed parameter space of models such as Random Forest and LightGBM [27].

Algorithms for building stacking models

Algorithm 2 displays the pseudocode for creating the model that is suggested in this study.

Where K represents the number of subsets when using the K-CV technique, P[DT] and P[LGBM] are lists to hold test results, Xi^{norm} represents independent features, yi represent dependent features, and Utrain represents the raw training dataset.

Algorithm 2

……

Input: Training data, $Utrain$ = $\{Xi^{norm}, yi\}_{i=1}^{m}$ where Xi^{norm} = independent features, and $yi \in y$ where y is dependent features

Output: NIDS

Begin

Step 1: adopt a cross-validation approach in preparing a training set for meta classifier

Randomly split $Utrain$ in K equal-size subsets: $Utrain$=$\{Utrain^1$,……$Utrain^K\}$

Step 2: for i = 1 to K

Make validation subset: VU$train$= U$train^i$

Make training subset: TU$train$= U$train$- U$train^i$

Using TU$train$ to train base classifiers (DT, LGBM) in parallel

Using VU$train$ to test base classifiers , and get result $\{ai^i$……$aj^i\}$ where j is validation row length

P[DT] or P[LGBM] Append $\{ai^i$……$aj^i\}$ in parallel

Step 3: Constructing a new train set M={P[DT], P[LGBM], y}

Step 4: Training meta classifier (RF)

Step 5: Retraining base classifiers using U$train$

Experimental Setup and Hyperparameter Optimization

Performance evaluation of the stacking intrusion detection model was performed using Python machine learning packages like Pandas, NumPy, and scikit-learn on a 64-bit Windows 10 Education operating system. To make it more relevant to resource-limited settings, the tests were done on regular hardware with an Intel Core i3 processor and 4 GB RAM.

For obtaining optimal hyperparameters for each tree-based base classifier, Bayesian optimization was employed. Parameter intervals and optimized parameter values for the base classifiers are listed in Table 3.

Table 3: Hyperparameters for the Classifiers

DT	criterion='gini', max_depth=14, min_samples_leaf=11, min_samples_split=7
LGBM	learning_rate=0.05586497233621369,max_depth=79,n_estimators=55
RF	criterion='gini',max_depth=118,min_samples_leaf=4,min_samples_split=1,random_state=101

Evaluation Metrics

To evaluate the predictive performance and practical effectiveness of the proposed stacking model, its results are compared with state-of-the-art approaches using standard intrusion detection metrics derived from the confusion matrix [29].

The performance metrics are defined as follows:

A. **Accuracy: Accuracy** is the proportion of correctly classified instances (true positives and true negatives) among all instances.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

B. **Precision: Precision** (positive predictive value) measures the proportion of correctly predicted positive instances out of all predicted positives.

$$\text{Precision} = TP / (TP + FP)$$

C. **Recall:** Recall (sensitivity or true positive rate) measures the proportion of correctly identified positive instances out of all actual positives.

$$\text{Recall} = TP / (TP + FN)$$

D. **F1-score:** F1-score is the harmonic mean of precision and recall, providing a balanced measure between them.

$$\text{F1-score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

E. **FAR (False Alarm Rate): FAR** represents the proportion of false positives among all negative instances.

$$\text{FAR} = FP / (FP + TN)$$

CINEFORUM
ISSN: 0009-7039
Vol. 66. No. 2, 2026

Intrusion Detection Performance and Results

To evaluate the predictive validity of the framework, the UNSW-NB15 dataset was partitioned into an 80% training set and a 20% independent testing set. A 5-fold cross-validation protocol was applied within the training pool to isolate the optimal hyperparameters before final model compilation and evaluation on the held-out test data.

Table 4 contrasts the multi-class verification results yielded by the isolated base estimators against the integrated meta-learning framework. On the UNSW-NB15 dataset, the proposed stacking model achieves an **Accuracy of 82.36%**, **Precision of 83.61%**, **Recall of 82.36%**, an **F1-Score of 79.87%**, and a **False Alarm Rate (FAR) of 0.0023**.

Table 4: Proposed model's detection results classifier

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	FAR	Time of Execution (s)
DT	80.20	80.42	80.20	79.91	0.0022	0.87
LGBM	81.40	81.00	81.40	79.42	0.0023	4.17
Stacking(My Approach)	82.36	83.61	82.36	79.87	0.0023	34.4

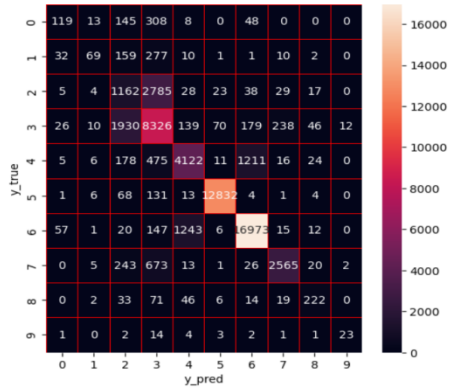


Fig. 5: Confusion matrix of Decision Tree

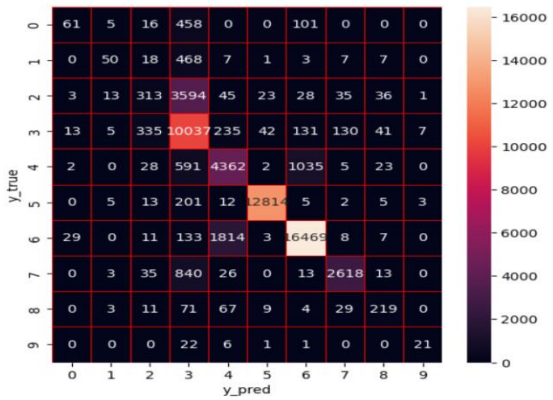


Fig. 6: Confusion matrix of LightGBM

CINEFORUM
ISSN: 0009-7039
Vol. 66. No. 2, 202

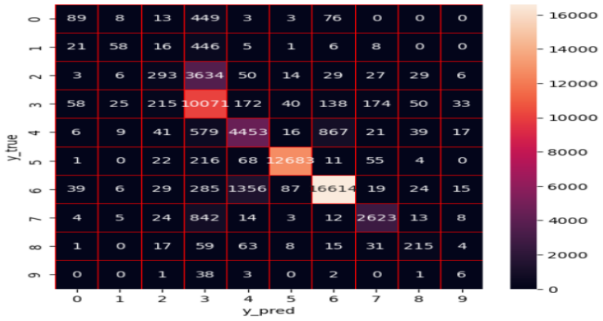


Fig. 7: Confusion matrix of Stacking model

Performance comparison with another classifier

Table 5 presents the performance comparison of LR, SVM, RF, and KNN models on the test dataset using accuracy, precision, recall, F1-score, FAR, and execution time, along with confusion matrices shown in Figures 8–11. The results are summarized as follows:

- LR: Accuracy 73.37%, Precision 72.04%, Recall 73.27%, F1-Score 71.75%, FAR 0.0000, Time 9.24s
- SVM: Accuracy 74.31%, Precision 74.48%, Recall 74.31%, F1-Score 73.08%, FAR 0.0000, Time 1153.31s
- RF: Accuracy 81.77%, Precision 82.23%, Recall 82.60%, F1-Score 79.54%, FAR 0.0010, Time 20.52s
- KNN: Accuracy 78.30%, Precision 79.67%, Recall 79.67%, F1-Score 78.30%, FAR 0.0029, Time 102.94s
- Stacking: Accuracy 82.36%, Precision 83.61%, Recall 82.36%, F1-Score 79.87%, FAR 0.0023, Time 49.4s

Overall, LR shows the lowest precision on the UNSW-NB15 dataset, while RF performs better than individual models. However, the stacking model achieves the best overall performance across most evaluation metrics**.**

Table 5 : comparison with other classifier

Method	Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)	FAR	Time of execution(S)
LogisticRegression	73.27	72.04	73.27	71.75	0.0000	9.19
SVM	74.31	74.48	74.31	73.08	0.0000	1153.31
Romdam	81.77	82.23	82.6	79.54	0.0010	20.52

Forest						
KNN	78.30	79.67	78.30	78.30	0.0029	102.94
Stacking	82.36	83.61	82.36	79.87	0.0023	49.4

Conclusion and Future Work

This thesis presented a hybrid tree-based stacking ensemble framework to secure Advanced Metering Infrastructure (AMI) networks in smart grids against cyberattacks. By integrating an Information Gain Ratio feature selection technique, the original dataset was optimized down to the top 21 high-variance dimensions. The stacking model combined the predictive capabilities of three base classifiers: Decision Trees (DT), LightGBM (LGBM), and Random Forests (RF).

Empirical evaluation on the UNSW-NB15 dataset demonstrated the framework's strong performance, achieving an **Accuracy of 82.36%**, **Precision of 83.61%**, **Recall of 82.36%**, **F1-Score of 79.87%**, and a low **False Alarm Rate (FAR) of 0.0023**. The Random Forest base classifier also performed effectively on its own, yielding an accuracy of 81.77% and a FAR of 0.0010.

These results validate the effectiveness of the proposed hybrid model in accurately identifying network anomalies while keeping false alerts to a minimum. Future work will investigate deep learning methodologies to further improve sequential attack detection and overall system performance.

Reference

[1] M. Kuzlu, M. Pipattanasomporn, and S. Rahman, "Communication network requirements for major smart grid applications in HAN, NAN and WAN," *Computer Networks*, vol. 67, pp. 74–88, 2014.

[2] F. E. Abrahamsen, Y. Ai, and M. Cheffena, "Communication Technologies for Smart Grid: A Comprehensive Survey," *Sensors*, vol. 21, no. 23, p. 8087, 2021.

[3]Vijayanand, R., D. Devaraj, and B. Kannapiran. "Support vector machine based intrusion detection system with reduced input features for advanced metering infrastructure of smart grid." *2017 4th International conference on advanced computing and communication systems (ICACCS)*. IEEE, 2017.

[4]Swetha, R. Bala Sri, and K. Goklia Meena. "Smart grid-A network-based intrusion detection system." *International Journal of Computer Applications* 975 (2015): 8887.

[5]Khan, Suleman, et al. "Intelligent intrusion detection system in smart grid using computational intelligence and machine learning." *Transactions on Emerging Telecommunications Technologies* 32.6 (2021): e4062.
[6]Islam, Md Khairul, et al. "Network anomaly detection using lightgbm: A gradient boosting classifier." *2020 30th International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE, 2020.
[7]Rashid, Md Mamunur, et al. "Cyberattacks detection in iot-based smart city applications using machine learning techniques." *International Journal of environmental research and public health* 17.24 (2020): 9347.
[8]Rashid, Md Mamunur, et al. "Performance enhancement of intrusion detection system using bagging ensemble technique with feature selection." *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. IEEE, 2020.
[9]Fosić, I., D. Žagar, and K. Grgić. "Network traffic verification based on a public dataset for IDS systems and machine learning classification algorithms." *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*. IEEE, 2022.
[10] C. Axelsson, "The Base-Rate Fallacy and the Difficulty of Intrusion Detection", ACM Transactions on Information and System Security, 2000
[11] S. Garcia et al., "Anomaly Detection: A Survey", ACM Computing Surveys, 2009
[12] M. Aljawarneh et al., "Intrusion Detection Systems: A Comprehensive Review", Journal of Network and Computer Applications, 2020
[13]G. Bontempi et al., "Machine Learning Strategies for Multi-class Support Vector Machines Classification," Applied Soft Computing, vol. 7, no. 1, pp. 49-60, 2007.
[14] J. R. Jang and N. K. Kim, "An Effective Intrusion Detection Method Using Feature Selection and Enhanced Majority Voting of Multiple Classifier Systems," IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews, vol. 42, no. 6, pp. 1341-1352, 2012.
[15] N. K. Kasabov, "Hybrid Intelligent Systems for Time Series Prediction," Applied Soft Computing, vol. 11, no. 2, pp. 2290-2302, 2011.
[16] A.Gerkis, J.Purcell. A survey of wireless mesh networking security, technology and threats. SANS institute infosec reading room, 2006.
[17] Moustafa, N., & Slay, J. (2015). "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." *Military Communications and Information Systems Conference (MilCIS)*.
[18] Moustafa, N., & Slay, J. (2016). "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set." *Information Security Journal: A Global Perspective*.
[19] Hancock, J. T., & Khoshgoftaar, T. M. (2020). "Survey on categorical data for neural networks." *Journal of Big Data*. *(Note: Standard citation for the structural comparison between label and one-hot encoding choices).*

[20]F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[21] Aljanabi, M., et al. (2021). "An analysis of feature selection methods for intrusion detection systems." *State of the Art in Cyber Security*.
[22] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.
[23] Zhou, Z. H. (2012). *Ensemble Methods: Foundations and Algorithms*. CRC Press.
[24] Breiman, L. (2001). "Random Forests." *Machine Learning*, 45(1), 5-32.
[25] Rajadurai, H., & Gandhi, U. D. (2020). "A stacked ensemble learning framework for intrusion detection in wireless networks." *Neural Computing and Applications*, 34, pp. 1-15.
[26] Berrar, D. (2019). "Cross-Validation." *Encyclopedia of Bioinformatics and Computational Biology*, 1, pp. 542-545.
[27] Snoek, J., Larochelle, H., & Adams, R. P. (2012). "Practical Bayesian optimization of machine learning algorithms." *NeurIPS*, 25, pp. 2951-2959.
[28] Wolpert, D. H., & Macready, W. G. (1997). "No free lunch theorems for optimization." *IEEE Transactions on Evolutionary Computation*, 1(1), pp. 67-82.
[29] Powers, D. M. (2020). "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation." *Journal of Machine Learning Technologies*.