CROSS-DATASET GENERALIZATION OF INTRUSION DETECTION SYSTEMS USING RANDOM FOREST, XGBOOST, LOGISTIC REGRESSION AND LSTM AUTOENCODER

DSECS ZG628T DISSERTATION

by

Srinivasan AN
2023CS0410

Project Work carried out at

CISCO, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
Pilani (Rajasthan) India

September, 2025

CROSS-DATASET GENERALIZATION OF INTRUSION DETECTION SYSTEMS USING RANDOM FOREST, XGBOOST, LOGISTIC REGRESSION AND LSTM AUTOENCODER

Submitted in partial fulfilment of the requirements of

M. Tech. Data Science & Engineering Program

by

Srinivasan AN
2023CS0410

Under the supervision of

Vigneshwari Viswanathan, Software Engineering Technical Leader

Project Work carried out at

CISCO, Bangalore

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI (RAJASTHAN)

September, 2025

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

CERTIFICATE

This is to certify that the Project Work entitled **CROSS-DATASET GENERALIZATION OF INTRUSION DETECTION SYSTEMS USING RANDOM FOREST, XGBOOST, LOGISTIC REGRESSION AND LSTM AUTOENCODER** and submitted by **Srinivasan AN** ID No. **2023cs0410** in partial fulfillment of the requirements of DSECS ZG628T Dissertation, embodies the work done by him/her under my supervision.

Vigneshwari (Sep 12, 2025 20:56:47 GMT+5.5)

Date: 12/09/2025

Signature of the Supervisor
Name: Vigneshwari Viswanthan
Designation: Software Engineering Technical Leader

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned my efforts with success.

Guidance and deadlines play a very important role in successful completion of the report on time. I wish to express my deep sense of gratitude to my Internal Guide, for able guidance and useful suggestions, which helped me in completing my project work, in time.I take immense pleasure in thanking **Vigneshwari Viswanathan & Prof.Seetha** who had been a source of inspiration and for timely guidance in the conduct of the work.

Finally a note of thanks to my classmates and the Department of Computer Science Engineering, both teaching and non-teaching staff for their co-operation extended to me.

Srinivasan AN

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
DSECS ZG628T **DISSERTATION**
SECOND SEMESTER 2024- 2025

Project Work Title : **CROSS-DATASET GENERALIZATION OF INTRUSION DETECTION SYSTEMS USING RANDOM FOREST, XGBOOST, LOGISTIC REGRESSION AND LSTM AUTOENCODER**

Name of Supervisor : Vigneshwari Viswanathan

Name of Student : Srinivasan AN

BITS ID No. of Student : 2023cs0410

Abstract *:*

Intrusion Detection Systems (IDS) are frequently benchmarked on individual datasets, where classification performance is optimized through algorithmic enhancements, ensemble methods, or extensive preprocessing. While these improvements often raise in-dataset accuracy, they do not necessarily translate into stronger real-world intrusion detection. In practice, IDS models frequently fail to generalize when exposed to unseen attack data from different environments due to dataset shift, limiting their robustness and applicability in dynamic networks.
This dissertation investigates both intra-dataset and cross-dataset intrusion detection with a focus on strong day-level performance and interpretable per-attack reporting. Specifically, the study leverages the CSE-CIC-IDS-2017 and CSE-CIC-IDS-2018 datasets to evaluate model generalization. For supervised learning, Random Forest, XGBoost, and Logistic Regression are applied to both intra-dataset (train and test on the same dataset) and cross-dataset (train on one year, test on the other) scenarios to measure their robustness under consistent vs. shifted distributions. In parallel, LSTM Autoencoder models are employed in an unsupervised setting to detect DoS/DDoS attacks, analyzing their ability to generalize across datasets while maintaining high detection performance and per-attack interpretability.

Key Words: Intrusion Detection systems, LSTM, Logistic Regression, Random Forest XGBoost, DOS/DDOS attacks, Day-Level Evaluation, Dataset shift, Cross-Dataset generalization, Intra-dataset evaluation

(Signature of Student)

Vigneshwari (Sep 12, 2025 20:56:47 GMT+5.5)
(Signature of Supervisor)

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY OF TERMS

AE	Auto Encoder
RE	Reconstruction Error
CM	Confusion Matrix
ML	Machine Learning
IDS	Intrusion Detection System
RF	Random Forest
Dos	Denial-of-Service
DDOS	Distributed Denial-of-Service

1. Introduction

Intrusion Detection Systems (IDS) play a critical role in modern cybersecurity by monitoring network traffic and identifying malicious activity. With the rapid evolution of cyberattacks, machine learning (ML) and deep learning (DL) techniques have become popular for building intelligent IDS solutions. While many studies report high accuracy when training and testing models on the same dataset, such results often fail to translate into real-world environments due to dataset shift and unseen attack patterns. This limitation highlights the importance of cross-dataset generalization—the ability of IDS models to maintain detection performance across different traffic sources and attack scenarios.

This dissertation investigates the generalization capability of supervised models (Random Forest, XGBoost, Logistic Regression) and an unsupervised deep learning model (LSTM AutoEncoder). By conducting both intra-dataset (training and testing within the same dataset) and cross-dataset (training on one dataset and testing on another) evaluations, the work aims to provide practical insights into how well IDS models adapt to new environments. Special attention is given to rare attack types such as Botnet and Infiltration, alongside volumetric attacks like DoS and DDoS, to assess detection robustness across diverse scenarios.

This Study comprehensively investigates the generalization capabilities of machine learning algorithms for network intrusion detection. The study evaluates models trained on one dataset (CIC-IDS-2017 or CSE-CIC-IDS-2018) and tested on another, focusing on both overall and per-attack performance.

1.1 Motivation

In real-world enterprise environments, IDS models are rarely deployed in the same conditions as the datasets on which they are trained. Traffic distributions, attack signatures, and benign behavior patterns vary significantly across organizations, networks, and time periods. As a result, models that appear highly effective in controlled benchmarks often struggle to detect novel or shifted attack patterns in practice. This gap between research performance and operational robustness motivates the need for systematic cross-dataset evaluation. By studying how models generalize across CIC-IDS-2017 and CIC-IDS-2018 datasets, this dissertation aims to replicate real-world conditions where attack distributions evolve over time, ensuring that IDS models remain reliable and resilient when faced with previously unseen traffic.

1.2 Problem Statement

Although machine learning models for intrusion detection have achieved impressive results on benchmark datasets, their effectiveness in real-world deployment remains uncertain due to poor cross-dataset generalization. Models often overfit to dataset-specific features, achieving near-perfect accuracy within the same dataset but failing to detect attacks when exposed to traffic from different environments. Therefore, there is a pressing need to systematically evaluate and compare machine learning and deep learning approaches for IDS in both intra- and cross-dataset settings. This dissertation addresses this gap by analyzing the generalization performance of Random Forest, XGBoost, Logistic Regression, and LSTM AutoEncoder across CIC-IDS-2017 and CIC-IDS-2018 datasets.

Scope of Attacks Considered:

This study focuses primarily on **DoS and DDoS attacks** for both LSTM AutoEncoder and supervised machine learning models, as these volumetric attacks represent the most prevalent and disruptive threats in modern networks. Additionally, **Botnet attacks** were included in the supervised learning experiments due to their distinct traffic patterns and relevance in distributed attack campaigns. Although infiltration was initially considered, it consistently

yielded poor detection performance even in intra-dataset settings, suggesting that the available data for this class was insufficient or too imbalanced for meaningful generalization. Therefore, infiltration was excluded from the final cross-dataset evaluation to maintain a realistic and balanced analysis.

1.3 Literature Survey

Intrusion Detection Systems (IDS) have been widely studied using machine learning (ML) and deep learning (DL) methods. The following section summarizes key contributions from the literature, highlighting both traditional ML approaches and modern DL-based techniques.

Survey and Foundational Works

Buczak et al. emphasized that making global recommendations for IDS is impossible due to data and attack variability, suggesting that IDS design must be dataset- and context-specific [1]. Srinivas and Manivannan proposed a deep learning method using Deep Belief Networks (DBN) to prevent Hello flood attacks in medical IoT networks [3].Malliga et al. reviewed deep learning methods for detecting DoS/DDoS attacks, highlighting evolving attack patterns and gaps in adaptive detection techniques [4].Ali et al. studied the effectiveness of combining BERT with SMOTE and an MLP classifier to enhance IDS classification. Their approach integrates the interpretability and context awareness of BERT with the efficient classification of MLP, improving performance on imbalanced network traffic [6].

Machine Learning with Modern Datasets

Zhang et al. developed a CNN-based IDS that integrates SMOTE with Gaussian Mixture Model (GMM) under-sampling to better handle class imbalance [8]. Joloudari et al. proposed a CNN-based model combined with SMOTE to address class imbalance in binary intrusion detection datasets [7]. Chimphlee et al. investigated the CSE-CIC-IDS-2018 dataset, applying preprocessing, feature selection, and seven classifiers (including MLP and XGBoost). Their results showed MLP performed best overall [5]

CHAPTER 2

Datasets and Preprocessing

2.1 Datasets

CIC-IDS-2017 and CSE-CIC-IDS-2018 were used as the primary evaluation datasets.

For the **LSTM Autoencoder**, the focus was narrowed to **Benign, DoS, and DDoS** labels. Other classes (e.g., Web, Botnet, Infiltration) were excluded to maintain emphasis on volumetric attacks and simplify cross-dataset comparisons.

For the **supervised algorithms (RF, XGBoost, LR)**, the scope included **DoS, DDoS, and Botnet** in addition to **Benign**.

Infiltration was retained only for **intra-dataset evaluation**, where it consistently showed weak detection performance. However, it was **excluded from cross-dataset experiments** due to poor generalization and lack of sufficient representative samples.

2.2 Feature Set

Features were extracted using CICFlowMeter, which outputs over 80 bidirectional flow-level statistics (forward and backward). These include:

- **Flow Basics**: Duration, total forward/backward packets and byte counts, flow bytes per second, flow packets per second.
- **Packet Size Stats**: Min, Max, Mean, Std, Variance for both forward and backward directions.
- **Inter-Arrival Times (IAT)**: Overall and directional IAT statistics—Mean, Std, Min, Max, plus totals.
- **TCP Flags Counts**: Frequency counts for PSH, URG, FIN, SYN, RST, ACK, CWR, ECE in both directions.
- **Header and Bulk Features**: Header lengths (forward/backward), bulk bytes/packets/rates for both directions, ratios.
- **Subflow Metrics**: Subflow forward/backward packets and bytes.

- **Window Size & Payload**: Initial window bytes (forward/backward), active and idle time features (Min, Mean, Max, Std), Act_data_pkt_forward, min segment size forward, average packet and segment size metrics.
- **Identifiers**: FlowID, source/destination IPs, ports, protocol. These are excluded prior to modeling to prevent data leakage.

2.3 Preprocessing

- Enabled correct coercion of numerical features to appropriate types (int64/float64).
- INF values were replaced with NaN. Any row containing NaN was dropped, avoiding imputation biases.
- Exact duplicate rows were identified and removed to avoid bias and wasted resources.
- Features with only a single unique value were removed, reducing dimensionality without losing information. It helped Reduced features from ~80 to ~68 (depending on dataset).
- Used Custom `df_shrink` function .It downcast numerical columns to smaller datatypes. This function intelligently downcasts numerical columns to the smallest possible data types (e.g., `int64 to int32 or float64 to float32`) without compromising the precision required by the data's range
- Flow ID, IP addresses, and source ports dropped due to high cardinality
- Timestamp features dropped as they added no extra predictive value
- Label harmonization: raw -> granular -> broad labels (e.g., “DoS Hulk”) into standardized granular labels (e.g., “DoS”).
- Memory optimization and persistent storage in Parquet
- After performing initial preprocessing steps on individual DataFrames for each dataset, they are combined into a single unified DataFrame for consistency and further processing.
- Feature alignment and Z-score normalization and Stripping whitespace from column names for consistency
- Robust handling of problematic characters and encoding issues within label strings to ensure clean and consistent labels.

2.3.1 Class Distribution after Preprocessing

CIC-IDS-2017

- **Benign:** 2,146,901
- **DoS:** 193,745
- **DDoS:** 128,014
- **PortScan:** 90,694
- **FTP-BruteForce:** 5,931
- **SSH-BruteForce:** 3,219
- **Botnet:** 1,948
- **Web Attack – Brute Force:** 1,470
- **Web Attack – XSS:** 652
- **Infiltration:** 36
- **Web Attack – SQL Injection:** 21
- **Heartbleed:** 11

CIC-IDS-2017

- **Benign:** 5,759,910
- **DDoS:** 775,955
- **DoS:** 196,568
- **Botnet:** 144,535
- **Infiltration:** 143,952
- **SSH-BruteForce:** 94,048
- **Web Attack – Brute Force:** 570
- **Web Attack – XSS:** 229
- **Web Attack – SQL Injection:** 85
- **FTP-BruteForce:** 53

Observation:

Both datasets exhibit **heavy class imbalance**, with Benign traffic dominating, followed by volumetric attacks (DoS/DDoS). Rare classes such as **Infiltration, Heartbleed, and Web/FTP attacks** have very few samples, making them unsuitable for cross-dataset generalization. This imbalance strongly motivated:

- The focus on **DoS, DDoS, and Botnet** for supervised models.
- The focus on **Benign, DoS, and DDoS** for LSTM Autoencoder comparisons.
- The exclusion of extremely rare classes from experiments to avoid misleading evaluations.

Harmonized Label Distribution

To enable consistent evaluation across datasets, granular classes were mapped to a **common set of broad labels** (Benign, DoS, DDoS, Botnet, Infiltration). Rare classes (e.g., Heartbleed, Web/FTP brute force) were excluded due to their extremely low representation.

Dataset	*Benign*	*DoS*	*DDoS*	*Botnet*	*Infiltration*
CIC-IDS-2017	2,146,901	193,745	128,014	1,948	36
CIC-IDS-2018	5,759,910	196,568	775,955	144,535	143,952

Observation:

- **Benign** traffic dominates both datasets.
- **Volumetric attacks (DoS/DDoS)** are well-represented and form the primary focus for cross-dataset generalization.
- **Botnet** has moderate representation in 2018 but is extremely rare in 2017.
- **Infiltration** is highly underrepresented in both datasets, leading to its exclusion from cross-dataset evaluation.

2.3.2 Data Splitting & Balancing Strategy (Supervised ML Models)

Train-test splits designed for both **intra-dataset** and **cross-dataset** scenarios.

Rebalancing performed using **undersampling + oversampling** with **class weights** to counter dataset imbalance.

HPO Rebalancing Parameters (subset for hyperparameter search):

- Function: rebalance_attacks
- min_per_attack=2000 → oversample rare attack classes if below threshold
- max_per_attack=50000 → downsample abundant classes if above threshold
- benign_ratio=1.0 → match benign samples to total attacks

Final Models Retrained on the full dataset (with dataset-specific parameters) for robustness:

- REBALANCING_PARAMS = { 'CIC_IDS_2017': {'min_per_attack': 1500, 'max_per_attack': 150000, 'benign_ratio': 0.5}, 'CIC_IDS_2018': {'min_per_attack': 50000, 'max_per_attack': 150000, 'benign_ratio': 1.0} }

Cross-Validation: K-Fold CV (5 folds) stratified on the HPO subset ensures rare attack classes are maintained during evaluation.

Randomized Search for HPO: Instead of grid search, **RandomizedSearchCV** was employed with up to **20 iterations** per model, balancing computational feasibility with robust hyperparameter exploration. The HPO subset size was capped at ~800,000 samples due to memory limits (16 GB RAM).

Custom refit metric for HPO: Introduced f1_weighted_attack_only (excludes Benign class) as the primary optimization target.This ensures hyperparameter tuning prioritizes attack detection instead of being dominated by the large Benign class.

Random Seeds were fixed across libraries (NumPy, sklearn, XGBoost, RF) to guarantee reproducibility of splits and rebalancing.

Validation Choice Justification: 5-fold CV was selected as a trade-off between computational cost and ensuring sufficient representation of rare classes (e.g., Botnet in 2017).

Benign-to-Attack Ratio Rationale:

- CIC-IDS-2017: benign_ratio=0.5 to prevent benign dominance.
- CIC-IDS-2018: benign_ratio=1.0 to maintain equal weightage of benign and attack samples.

Leakage Prevention: Care taken to ensure stratified folds and train-test splits prevent leakage across flows, preserving temporal and class integrity.

2.3.3 Class Weighting for Final Model Training (Supervised ML Models)

After HPO, final models were trained on the **full dataset**, rebalanced using rebalance_attacks.

X_train_final_bal, y_train_final_bal formed the final balanced training sets.

Sample Weights applied selectively to handle residual imbalance in datasets with extremely rare classes.

Dataset-Specific Strategies:

CIC-IDS-2018: Sufficiently balanced after rebalancing → only physical resampling applied, no sample weights used.

CIC-IDS-2017 (and datasets with rare classes): A **hybrid approach** combining rebalancing + sample weights was applied.

Weight Computation Method: Weights were assigned as the inverse of class frequency (1 / frequency), normalized across all classes, ensuring minority classes received proportionally higher influence during training.

CHAPTER 3 Methodology

3.1 Models

- Random Forest, XGBoost, Logistic Regression, LSTM AutoEncoder
- Hyperparameter grids defined per model

3.1.1 Random Forest

Type: Ensemble, tree-based classifier.

Purpose: Classify network flows into attack categories (DoS, DDoS, Botnet, Infiltration) or benign.

Mechanism:

Builds multiple decision trees on bootstrapped subsets of training data.

Each tree votes for a class; majority vote decides final prediction.

Handles high-dimensional data and non-linear feature interactions well.

Hyperparameters Tuned: Number of trees, max depth, min samples per leaf, max features per split.

3.1.2 XGBoost

Type: Gradient Boosted Decision Trees.

Purpose: Same as RF, optimized for higher accuracy and handling imbalanced data.

Mechanism:

Sequentially builds trees, each correcting errors of previous trees.

Includes regularization (L1/L2) to prevent overfitting.

Can use weighted loss functions to handle rare attack classes.

Hyperparameters Tuned: Learning rate, max depth, number of estimators, gamma, subsample ratio, colsample_bytree.

3.1.3 Logistic Regression (LR)

Type: Linear model for classification.

Purpose: Baseline model for attack detection, interpretable feature importance.

Mechanism:

Computes probability of class using logistic function: $P(y= 1|X) = 1/(1+e^{-\beta x})$

Supports multiclass classification via one-vs-rest (OvR).

Can incorporate class weights to address imbalance.

Hyperparameters Tuned: Regularization strength (C), penalty type (L2), solver type,tolerance

3.1.4 HyperParameter Grids

Random Forest:

- n_estimators: [20, 50, 80]
- max_features: [‘sqrt’]
- min_samples_leaf: [1, 2, 4]
- min_samples_split: [2, 5, 10]
- max_depth: [20, 30, 40]

XGBoost:

- n_estimators: [100, 200]
- learning_rate: [0.05, 0.1]
- max_depth: [3, 5, 7]
- subsample: [0.7, 0.9]
- colsample_bytree: [0.7, 0.9]

Logistic Regression:

- C: [0.01, 1]
- penalty: [‘l2’]
- solver: [‘saga’]
- max_iter: [5000, 10000]
- tol: [0.005]

3.1.5 LSTM Autoencoder (AE):

Purpose: Unsupervised anomaly detection for volumetric attacks (DoS/DDoS). AE is trained only on benign traffic and flags deviations (high reconstruction error) as attacks.

3.1.5.1 Autoencoder Framework for Intrusion Detection

1) Introduction

An Autoencoder (AE) is a type of neural network designed to learn an identity mapping.

$$\hat{X} \approx X$$

Given an input X, the AE attempts to **reconstruct it at the output**. By introducing a **bottleneck layer**,the network is forced to learn a **compressed representation** that captures the most salient features of the input distribution. This property makes AEs suitable for **anomaly detection** when trained only on benign traffic, as anomalous data often produces higher reconstruction errors.

2) Encoding Stage (Dimensionality Reduction)

The **encoder** f_θ maps the input sequence $X \in R^K$, where $K << d$

Mathematically:

$$z=f_\theta(X)$$

For an LSTM Autoencoder, stacked LSTM layers capture temporal dependencies in the sequence.
Each timestep's features are processed recurrently, and the final hidden state encodes the entire sequence.

2.1 Bottleneck (Latent Space)
The **bottleneck layer** provides a compressed summary of the sequence:

$$X^= g_\Phi(Z) = g_\Phi(f_\theta(X))$$

For **benign traffic**, the encoder captures stable flow patterns.

For **attack traffic** (DoS, DDoS), the latent representation fails to encode these patterns accurately, resulting in **high reconstruction error**.

3) Decoding Stage (Reconstruction)

The **decoder** gΦ reconstructs the original sequence from the latent representation:

$$X^= g_\Phi(Z) = g_\Phi(f_\theta(X))$$

The decoder uses **LSTM layers in reverse**.

A **TimeDistributed dense layer** produces the output at each timestep.

If the input sequence is benign, the reconstruction X^ closely matches the original X.

4) Reconstruction Error (Anomaly Score)

The anomaly score is based on the difference between X and X^. For a sequence X_1:T of length T, Mean Squared Error (MSE) is used:

$$E(\mathbf{x}_{1:T}) = \frac{1}{T \cdot d} \sum_{t=1}^{T} \sum_{j=1}^{d} \left(x_{t,j} - \hat{x}_{t,j}\right)^2$$

Where:

$X_{t,j.}$ = feature j at timestep t in the input

$X^{\wedge}_{t,j}$ = corresponding reconstruction

L(.) = element-wise loss (MSE)

High reconstruction error indicates anomalous (attack) traffic.

AE Model Architecture

Figure1 :

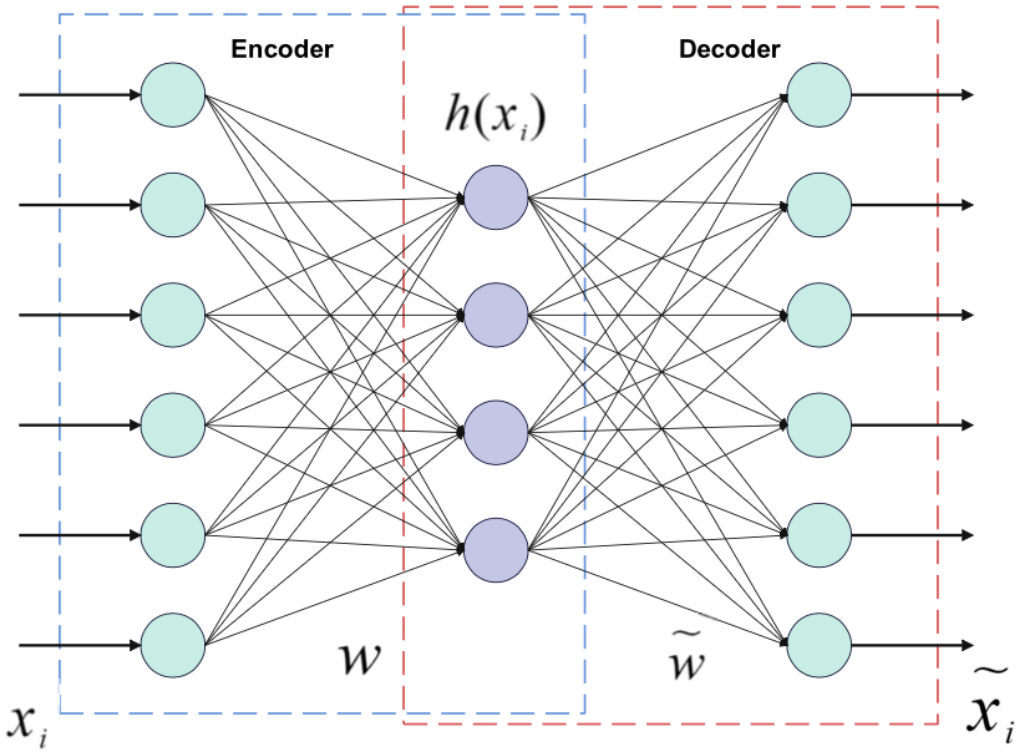
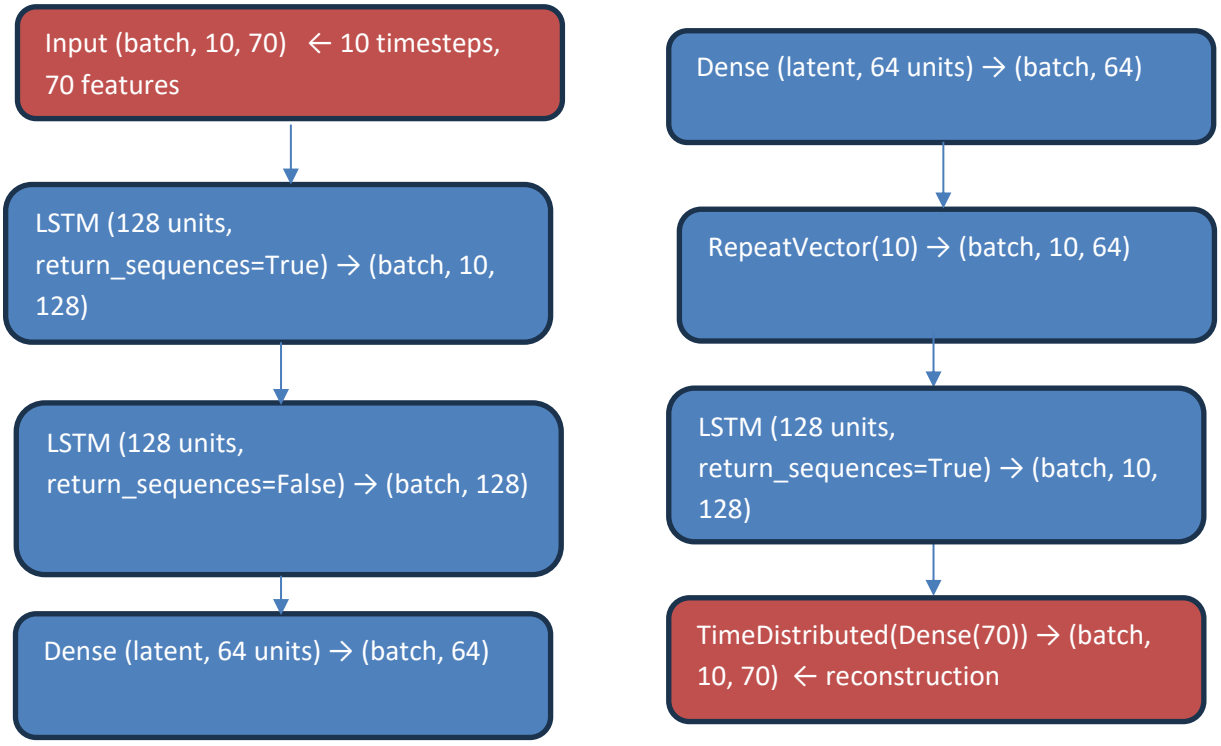


Figure2 : LSTM layer Dimension Flow Diagram



3.1.5.2 Threshold Selection for LSTM-AE (Global, Day, Per-Attack)

Computation Flow(Day-wise threshold) :

1) Compute reconstruction errors and labels

   err_day = compute_recon_error(ae, Xd_seq) → one error per sequence.

   y_true_day_overall = np.isin(yd_seq, [dos_id, ddos_id]).astype(int) → binary labels.

2) Compute raw day threshold (thr_day_raw)

If the day has **both benign and attack samples**, choose threshold from PR curve (default = maximize F1).

3) Clamp threshold to safe bounds:

Lower bound = Global floor → baseline derived from benign-only training errors.

4) Apply final threshold (thr_day):

y_pred = (err > thr_day).astype(int) for binary classification.

This ensures thresholds never drop below a globally safe floor (avoiding under-detection)
The result (thr_day) is the final threshold for detecting attacks on that day.

Global threshold (floor): Derived from benign-only training errors using a high quantile (e.g., 99–99.5%) and a robust estimate (median + k·MAD).

Day-level threshold (thr_day): Selected via PR curve (max-F1 by default) and The **final threshold for the day** must be at least as high as the global threshold.

Computation Flow (Per-attack) :-

1) Restrict to {benign + this attack type}.
2) Compute errors with a tailored metric.
3) Get a benign-anchored cap for that subset.
4) Choose supervised threshold (precision-target for DoS, F1 for DDoS).
5) Clamp it into a safe range: [global floor, min(day cap, benign cap, max error)].
6) Apply threshold for predictions and metrics.

3.1.6 Evaluation Metrics

Classification Metrics:

- **Accuracy:** Overall correct predictions.
- **Precision, Recall, F1-score:** Weighted and per-class; F1-score emphasized for imbalanced attack detection.
- **ROC-AUC:** Measures discrimination power of classifier between benign and attack classes.

Day-level Evaluation:

- Metrics computed per day to capture temporal variations in attack patterns.
- Confusion matrices generated for detailed per-attack analysis.

LSTM-AE Specific:

- **Reconstruction error (MSE)** used as anomaly score.
- **Threshold selection:** Global floor + day-level adjustment to detect attacks.
- Performance evaluated using F1-score, Accuracy, and ROC-AUC.

CHAPTER 4

Results

4.1 Intra-Dataset Evaluation

- Overall and per-attack metrics
    - Random Forest trained on CIC-IDS-2018 achieved **93.8% accuracy** and **95.3% weighted F1-score**, with near-perfect detection of DDoS and DoS, strong Botnet detection, and moderate Infiltration detection (F1≈0.35–0.62 across days).
    - Similarly, Random Forest on CIC-IDS-2017 achieved **99.9% accuracy** and **99.9% weighted F1-score**, showing almost perfect DDoS and DoS detection, high Botnet detection, and strong Infiltration detection (F1≈0.97 on the Infiltration day).
    - Logistic Regression, however, showed weaker intra-dataset results: on CIC-IDS-2018 it reached **51.3% accuracy** and **59.8% weighted F1-score**, with strong detection of DDoS (F1≈0.966) and Botnet (F1≈0.983) but weaker Infiltration performance; on CIC-IDS-2017 it achieved **52.8% accuracy** and **67.3% weighted F1-score**, with solid DDoS detection (F1≈0.761) and DoS detection (F1≈0.899) but poor performance on Infiltration and Botnet in some days.
    - XGBoost tained on CIC-IDS-2018 achieved perfect score for DOS,DDOS and Botnet detection and Poor Infiltration (Recall≈0.32–0.47 across days).
    - Similarly, XGBoost on CIC-IDS-2017 showing almost perfect DDoS and DoS detection, Botnet detection, and strong Improved detection (F1≈0.84 on the Infiltration day).

Fig3: IntraDataset Evaluation -2017

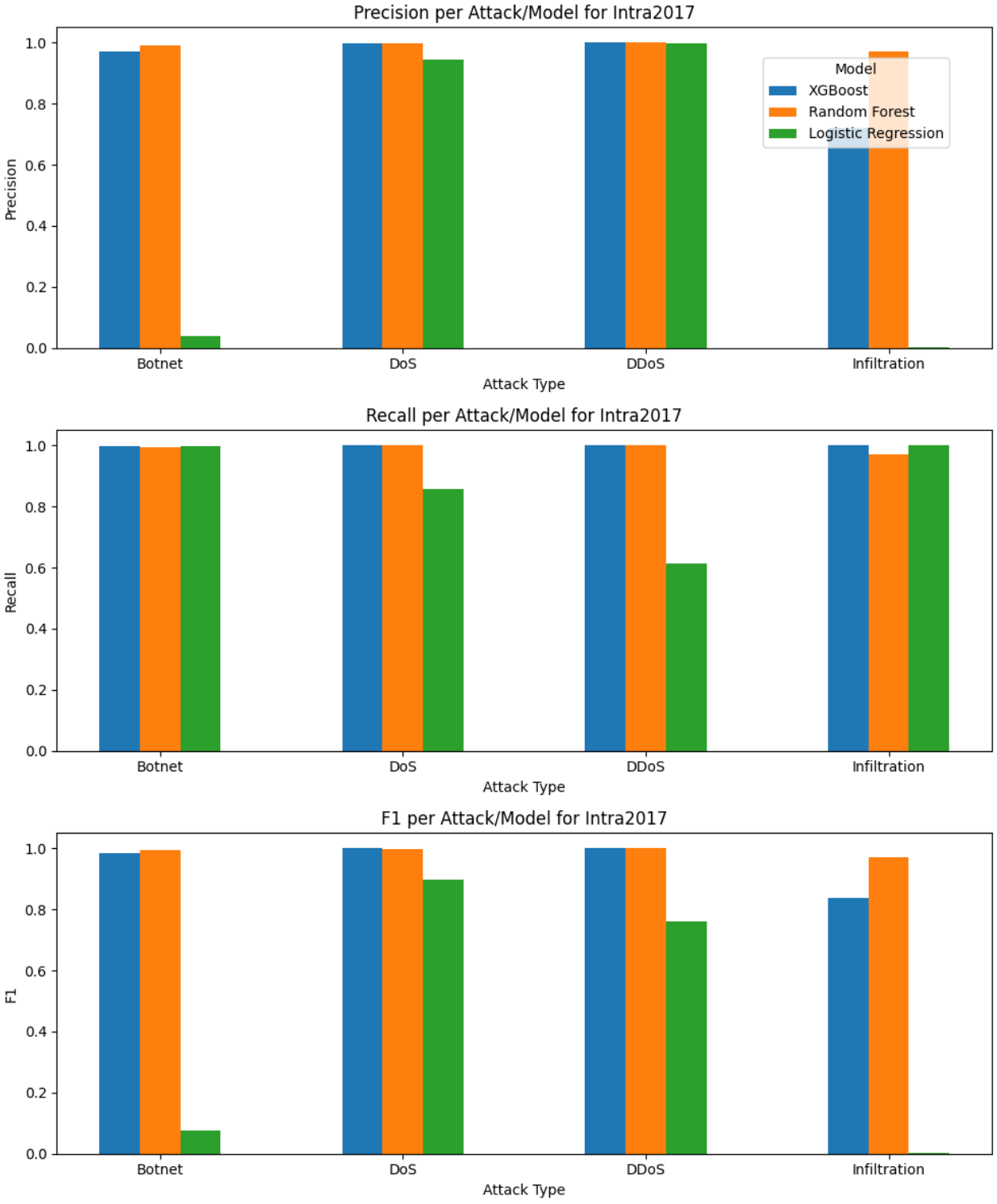
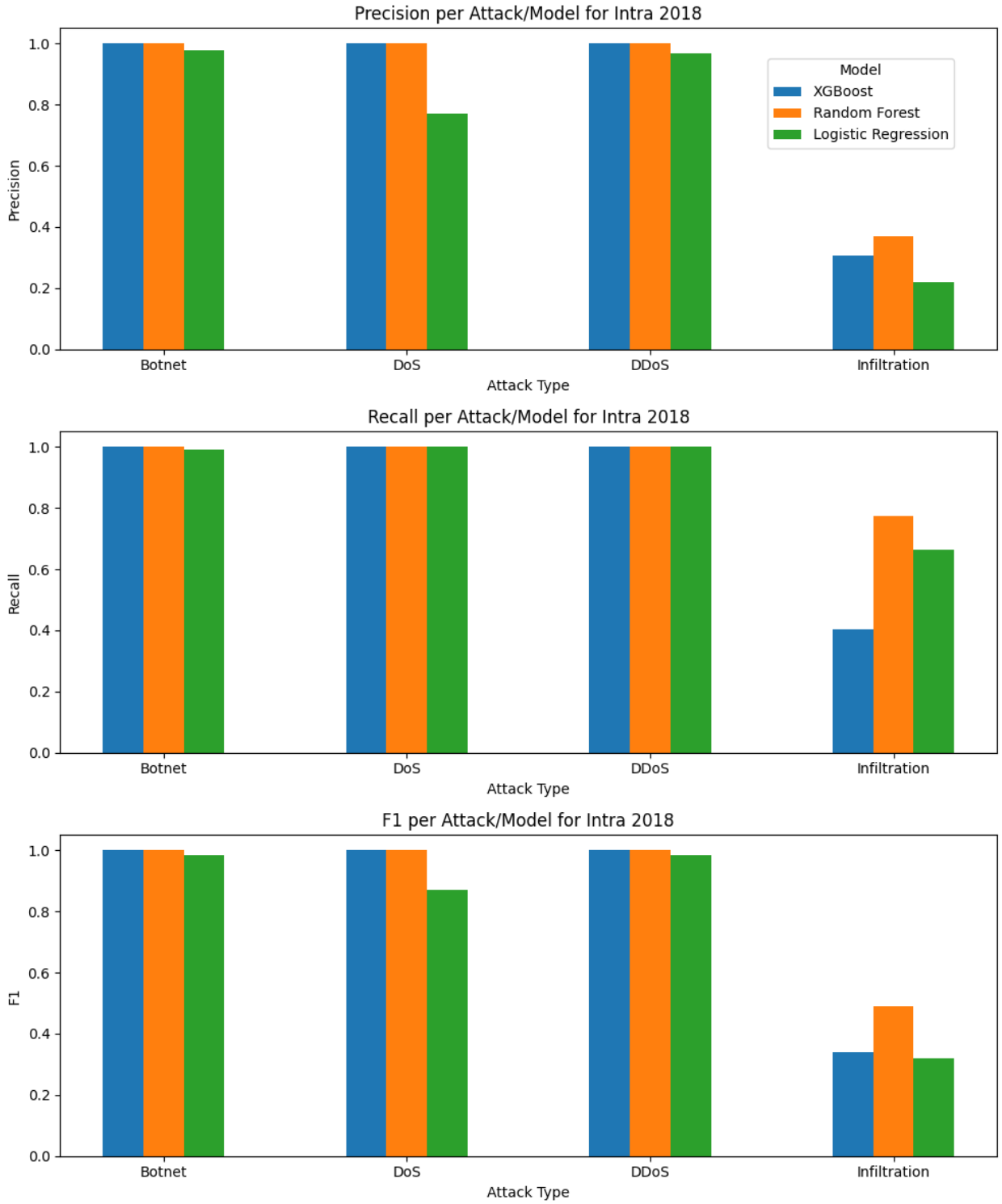


Figure4: IntraDataset Evaluation -2018



4.2 Cross-Dataset Evaluation (Supervised ML Models)

4.2.1 Train_CIC_IDS_2018_Test_CIC_IDS_2017

Per-day performance (XGBoost)

Botnet (Friday morning): Precision 0.964, Recall 0.264 → F1 ≈ 0.41

Strong precision, but model misses ~74% of Botnet cases.

DoS (Wednesday): Precision 0.975, Recall 0.285 → F1 ≈ 0.44

Again, high precision but low recall.

DDoS (Friday afternoon): Precision 0.419, Recall 0.0001 → F1 ≈ 0.0002

Model fails to detect DDoS almost entirely.

Per-Day Per Performance (Logistic Regression)

Botnet (1948 samples): Precision = 0.25, Recall = 0.0067 → almost all missed**.**

DDoS (128k samples): Very strong detection → Precision = 0.9775, Recall = 0.8847, F1 = 0.9287.

DoS (193k samples): Moderate → Precision = 0.4656, Recall = 0.7678, F1 = 0.5797

Logistic Regression detects **large-scale attacks like DDoS well**, has some ability on DoS, but struggles with **Botnet** and **Inflitration**

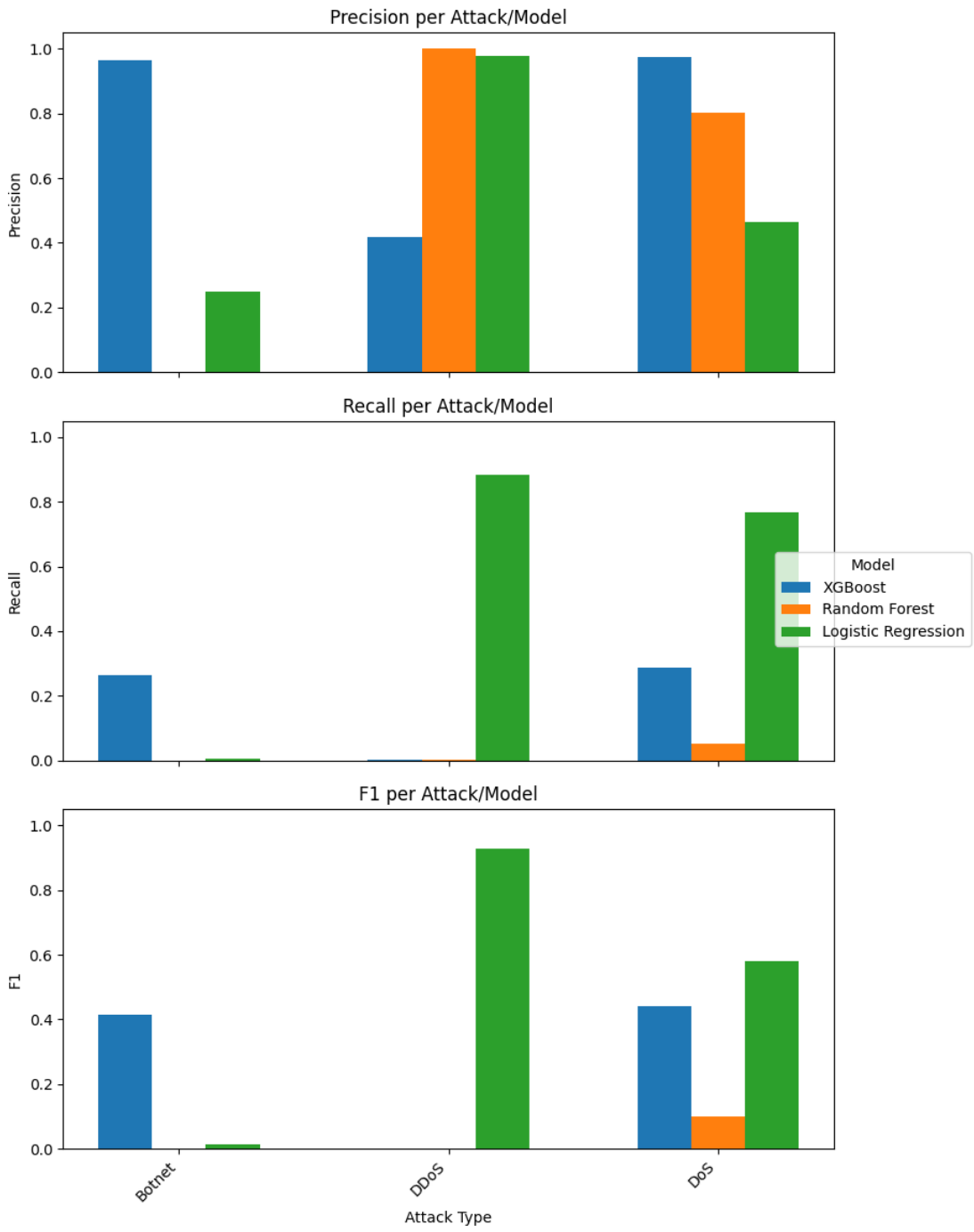
Per-day performance (Random Forest)

Botnet (1948 samples): Precision = 0.0, Recall = 0.0, F1 = 0.0 → completely missed.

DDoS (128k samples): Precision = 1.0, Recall ≈ 0.0001, F1 ≈ 0.0003 → classifier almost never predicted DDoS.

DoS (193k samples): Precision = 0.8029, Recall = 0.0529, F1 = 0.0993 → very low recall despite high precision.

Figure5:Model Performance (Train_2018_Test_2017)



4.2.2 Train_CIC_IDS_2017_Test_CIC_IDS_2018

Per-day performance (XGBoost)

DDoS1 (Tuesday): Precision 1.0, Recall 0.0014 → F1 ≈ 0.0029
Model predicts very few positives (high precision), but misses 99.9% of actual attacks.

Botnet (Friday): Precision 0.988, Recall 0.0082 → F1 ≈ 0.016
Extremely low recall; detects <1% of Botnet cases.

DoS2 (Friday): Precision 1.0, Recall 0.565 → F1 ≈ 0.722
Very strong performance here.

DoS1 (Thursday): Precision 0.982, Recall 0.475 → F1 ≈ 0.640
Decent, but recall still moderate.

DDoS2 (Wednesday): Precision 0.0, Recall 0.0 → complete failure

Per-day performance (RandomForest)

DDoS1 (575k samples): Precision = 0.0, Recall = 0.0, F1 = 0.0 → completely missed.

Botnet (144k samples): Precision = 1.0, Recall = 0.0, F1 ≈ 0.0001 → model never actually flagged positives.

DDoS2 (200k samples): Precision = 0.0, Recall = 0.0, F1 = 0.0 → completely missed.

DoS (145k samples, Friday): Precision = 1.0, Recall = 0.014, F1 = 0.0276 → extremely low recall.

DoS (51k samples, Thursday): Precision = 0.9932, Recall = 0.4719, F1 = 0.64 → best detection across all scenarios, but still poor recall.

Per-day performance (Logistic Regression)

DDoS1 (20 Feb 2018) (575K Samples)

Precision: **0.817** | Recall: **0.109** | F1: **0.193**

LR can identify some DDoS flows, but recall is very low (≈11%). It catches only a small portion of the attacks.

DDoS2 (21 Feb 2018) (200K samples)

Precision: **0.000** | Recall: **0.000** | F1: **0.000**

Complete failure → No detections.

Botnet (02 Mar 2018)(144k Samples)

Precision: **0.000** | Recall: **0.000** | F1: **0.000**

Model did not recognize any Botnet flows.

DoS1 (15 Feb 2018)(51k samples)

Precision: **0.134** | Recall: **0.164** | F1: **0.148**

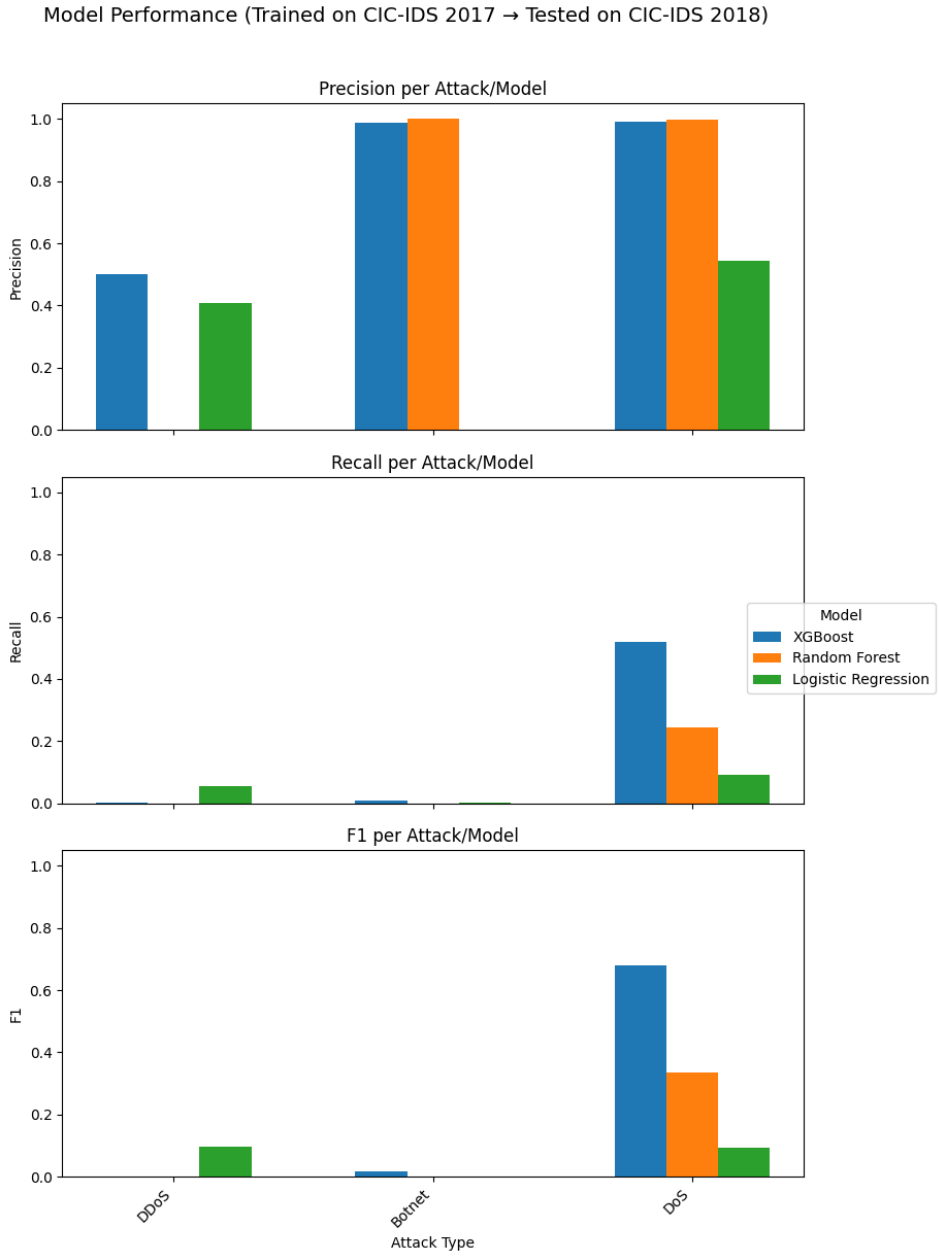
Very weak detection; low precision & recall. LR struggles to separate DoS1 from normal.

DoS2 (16 Feb 2018)(145k samples)

Precision: **0.957** | Recall: **0.018** | F1: **0.035**

Extremely skewed: it's *almost always wrong by omission*. When it predicts DoS, it's right (high precision), but it predicts almost nothing.

Figure6: Model Performance (Train_2017_ Test_2018)



Overall, cross-dataset generalization results were significantly weaker compared to intra-dataset evaluation. Models that achieved near-perfect detection within a dataset suffered large drops in accuracy and F1-scores when tested on another dataset.

Best cross-dataset performance: Logistic Regression (only when trained on 2018 → tested on 2017, largely due to strong DDoS detection).

Most consistent but still weak: XGBoost (moderate across both directions, but low attack-only F1).

Worst generalization: Random Forest (very high accuracy due to benign bias, but almost zero attack detection).

Models overfit to dataset-specific traffic patterns (packet distributions, feature ranges, background benign traffic) rather than truly learning attack signatures. cross-dataset results **demonstrate the sensitivity and fragility of ML-based IDS** — high scores within one dataset are not evidence of robustness. True IDS evaluation must test **generalization across environments and datasets**.

4.3 LSTM AE Cross-Dataset Results

The section presents the detailed cross-dataset evaluation results for the LSTM AutoEncoder (AE) model. The AE was trained on Benign samples from CIC-IDS-2018 and evaluated on CIC-IDS-2017 (and vice versa in later sections). Results are shown day-by-day, capturing both overall metrics and per-attack performance (restricted to DoS and DDoS).

Day: DDoS-Friday-WorkingHours-Afternoon (2017)

Overall: Accuracy = 0.7819, Precision = 0.7977, Recall = 0.8307, F1 = 0.8139, ROC-AUC = 0.7101 → solid performance.

DDoS-specific: F1 = 0.81, ROC-AUC ≈ 0.71 → good detection.

Thresholds: thr_day_raw = 4.572420, thr_used = 0.355752

Interpretation: This is a “good/easier” day for the model. Attack anomaly scores are high, benign scores remain low.

Figure7:RE Error(DDOS_Friday_workingHours)

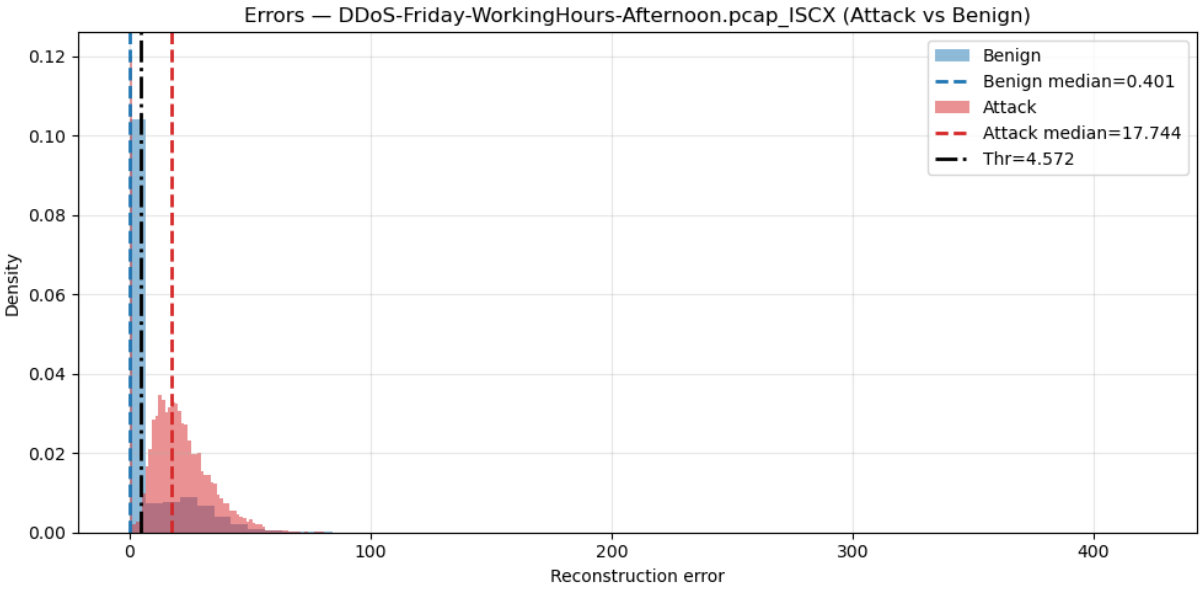


Figure8:ROC curve (DDOS_Friday_workingHours)

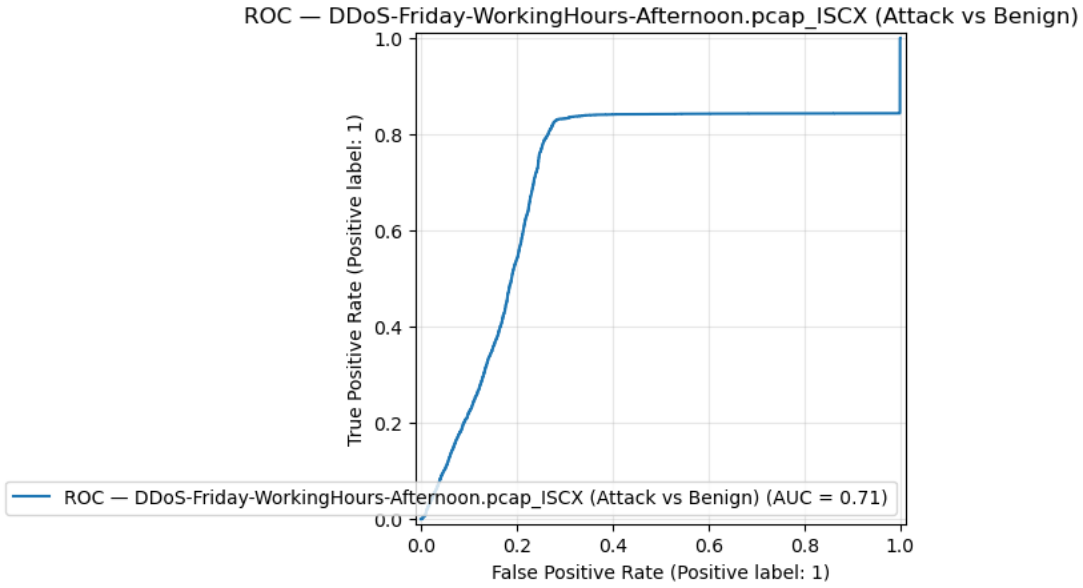
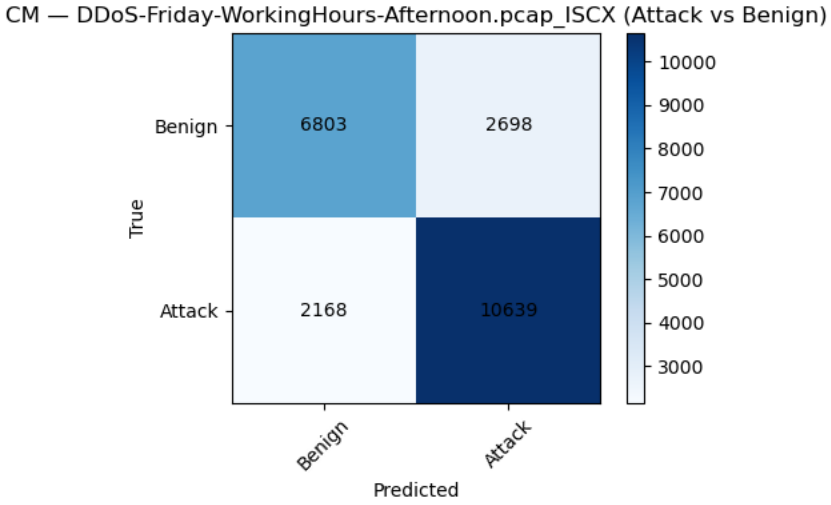


Figure9: CM- DDOS_Friday_workingHours



Day: DoS-Wednesday-WorkingHours (2017)

Overall: Accuracy = 0.91, Precision = 0.8956, Recall = 0.8289, F1 = 0.86, ROC-AUC = 0.92 → excellent performance.

DoS-specific: Strong separation between benign and attack flows.

Thresholds: thr_day_raw = 4. 544933, thr_used = 0.355752

Interpretation: Another "good/easier" day for the AE. Anomaly scores for DoS traffic are sharply higher than benign.

Figure10:RE Error(Dos_Wed_workingHours)

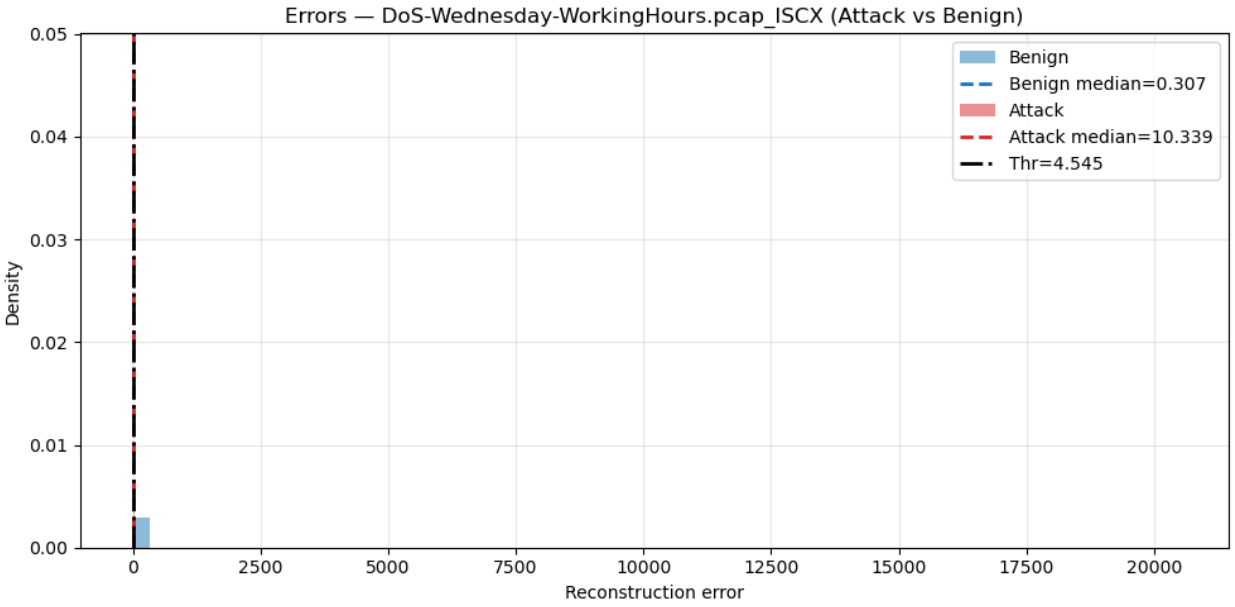


Figure11:ROC Curve(Dos_Wed_WorkingHours)

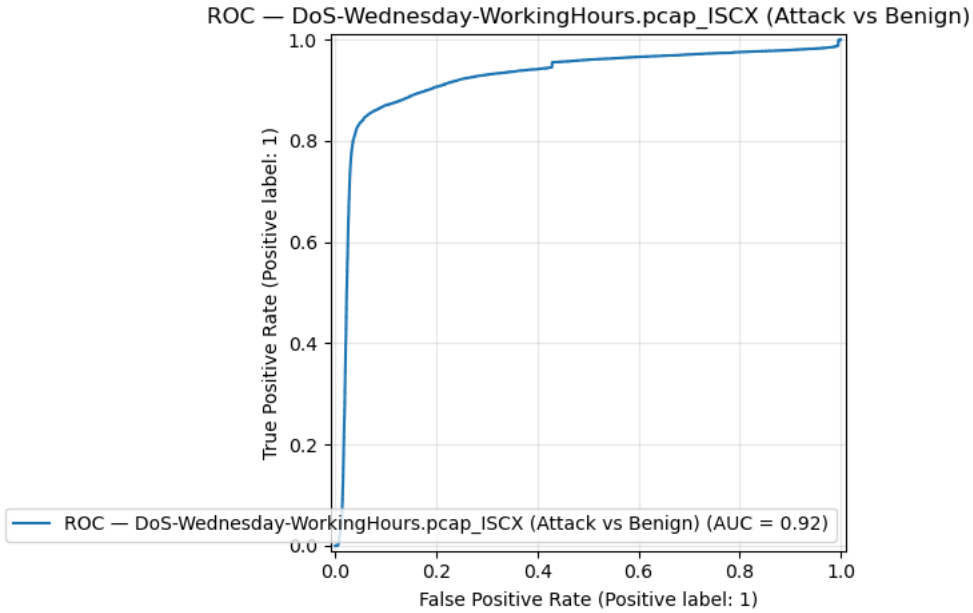
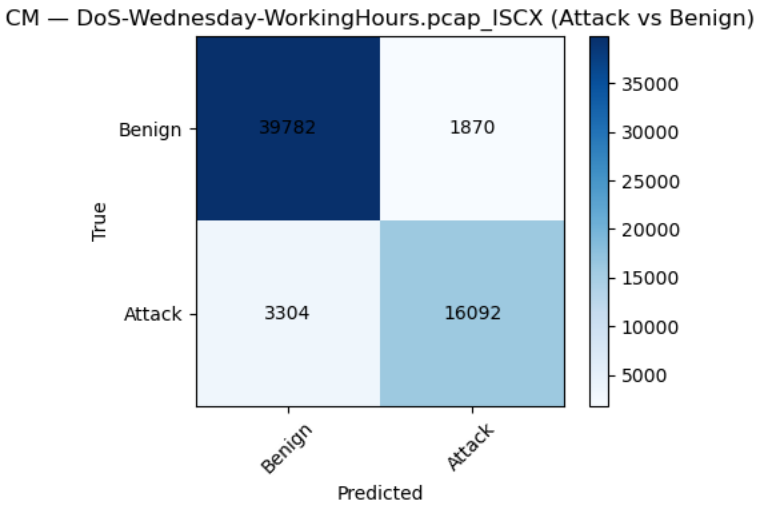


Figure12:CM(Dos_Wed_WorkingHours)



CIC-IDS-2018 Evaluation (Model trained on CIC-IDS-2017 Benign)

Day: DDoS1-Tuesday-20-02-2018

Overall: Accuracy = 0.6995, Precision = 0.7296, Recall = 0.7707, F1 = 0.7496, ROC-AUC = 0.7911

Thresholds: thr_day_raw = 0.282660, thr_used = 0.692385

Interpretation: Detection remains reasonable, but threshold adjustment was critical. A relatively harder day compared to CIC-IDS-2017 DDoS/DoS days. distribution drift made detection less stable

Figure13:RE Error (DDOS1-Tue)

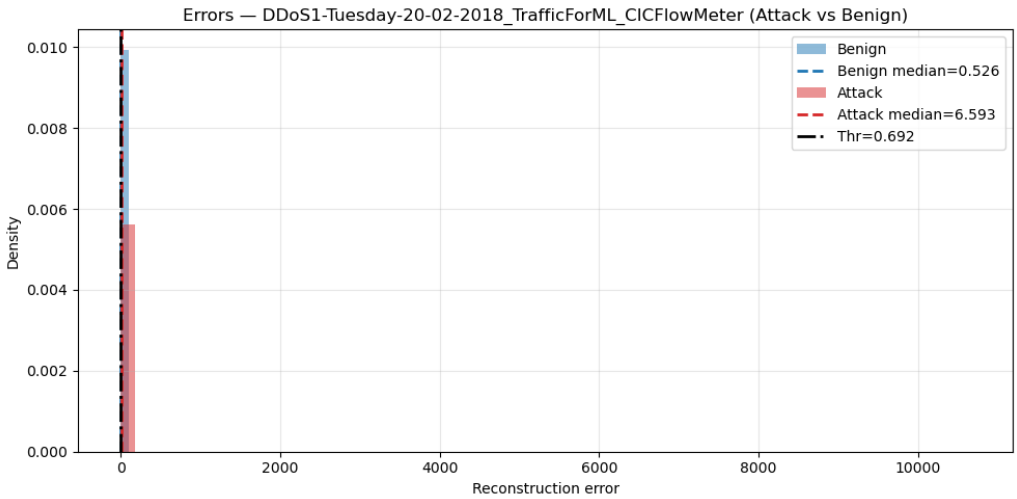


Figure14:ROC_DDOS1-Tue

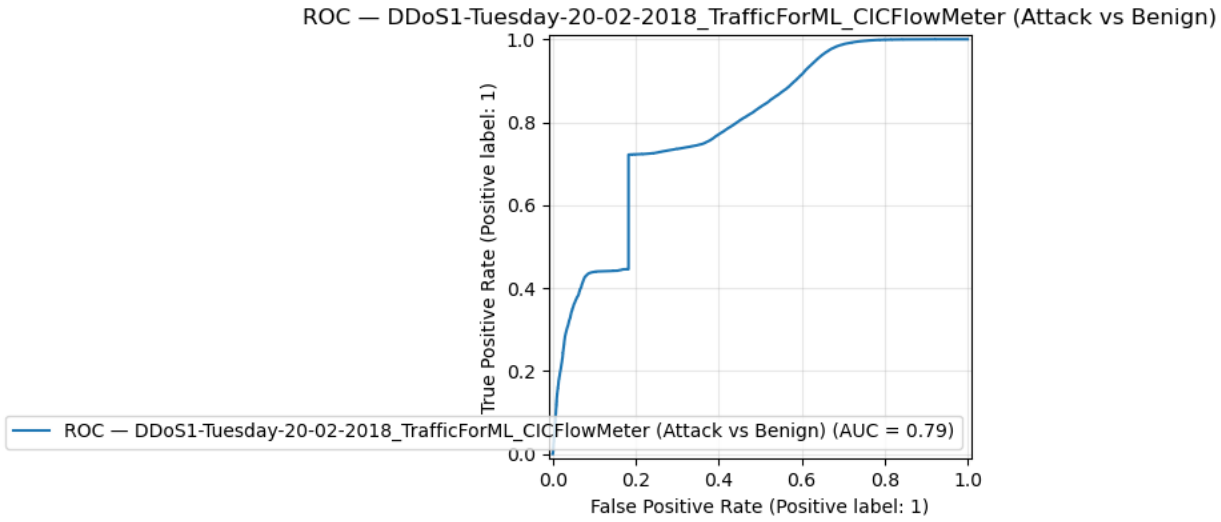
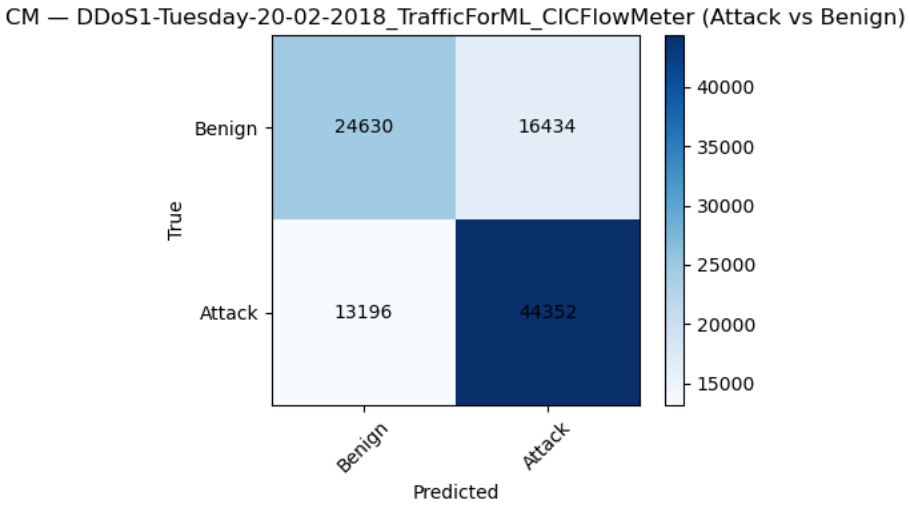


Figure15:CM_DDOS1_Tue



Day: DDoS2-Wednesday-21-02-2018

Overall: Accuracy = 0. 7857, Precision = 0. 7454, Recall = 0.6097, F1 = 0.6708, ROC-AUC = 0.5913

Thresholds: thr_day_raw = 6.850107 , thr_used = 0.692385

Interpretation: Detection remains Moderate, but threshold adjustment was critical. A relatively harder day compared to CIC-IDS-2017 DDoS/DoS days. distribution drift made detection less stable

Figure16:RE Error(DDOS2-Wed)

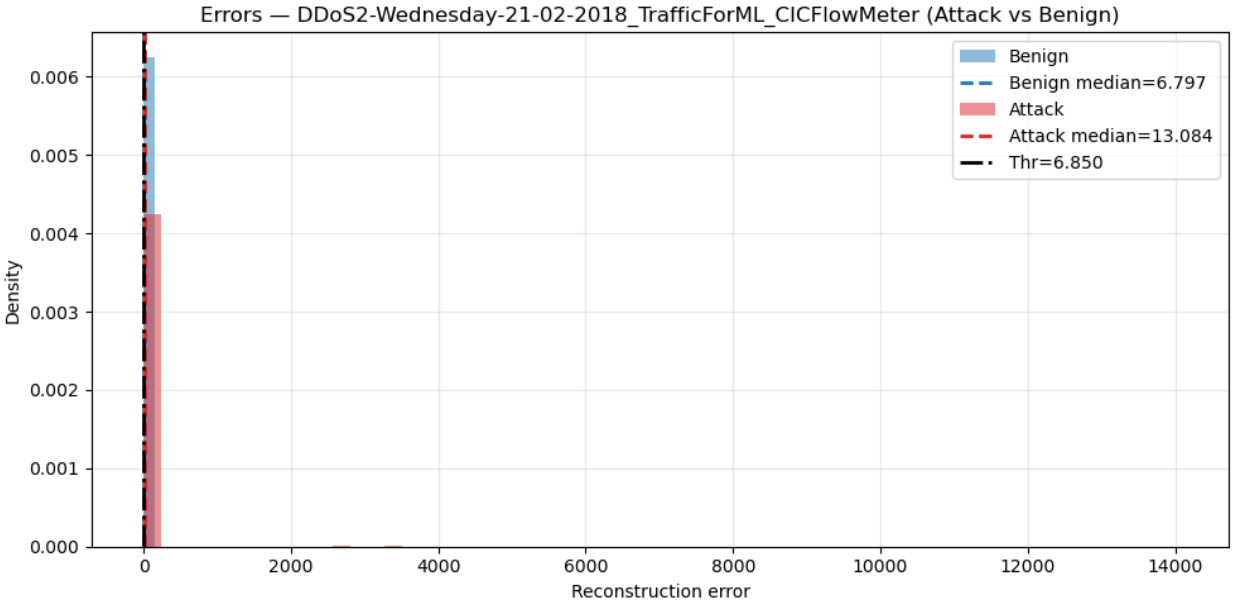


Figure16:ROC_ DDOS2-Wed

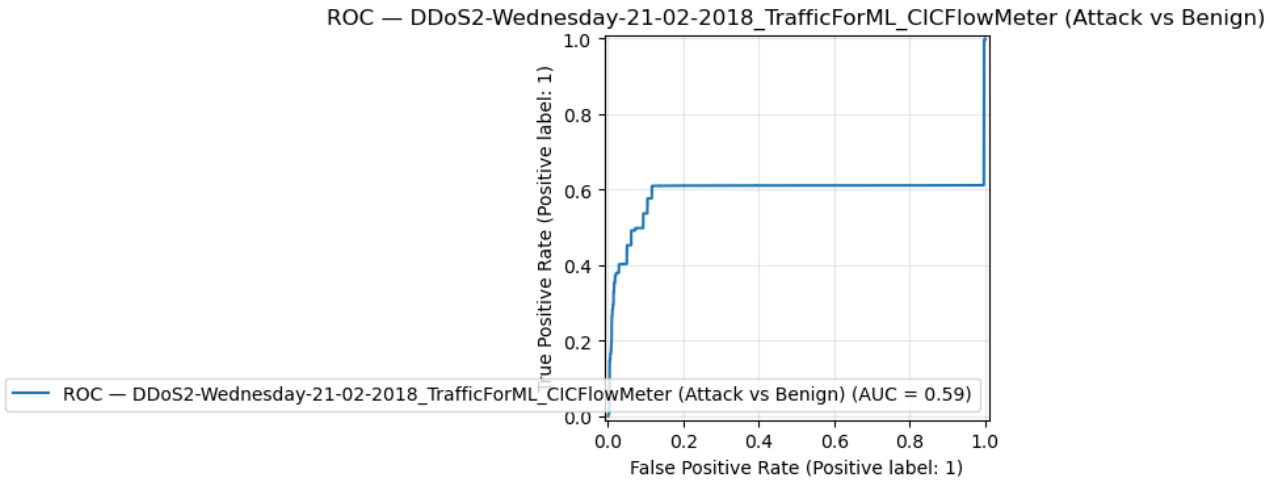
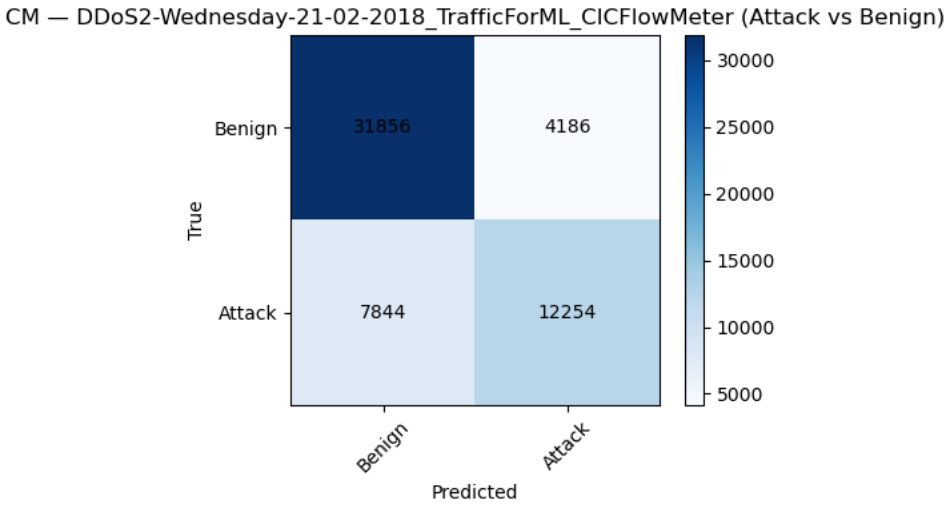


Figure18:CM_DDOS2-Wed



DoS1-Thursday-15-02-2018

Acc = 0.6295 | Prec = 0.0628 | Rec = 0.3834 | F1 = 0.1080 | ROC-AUC = 0.3615

thr_day_raw=0.010841 thr_used=0.692385

Interpretation: "Hard" day — partial attack detection, but accuracy and ROC-AUC collapse due to misalignment of anomaly scores.

The day is "tough": benign and attack scores overlap so much that the PR curve wants a very low threshold (0.010841) to get recall, which would massively increase benign false positives.

floor protects precision: The clamp prevents dropping below the global benign floor, so the final threshold is 0.6924. That's conservative; it will cut false positives but very likely drives recall lower

Figure19:RE_Error (Dos1-Thu)

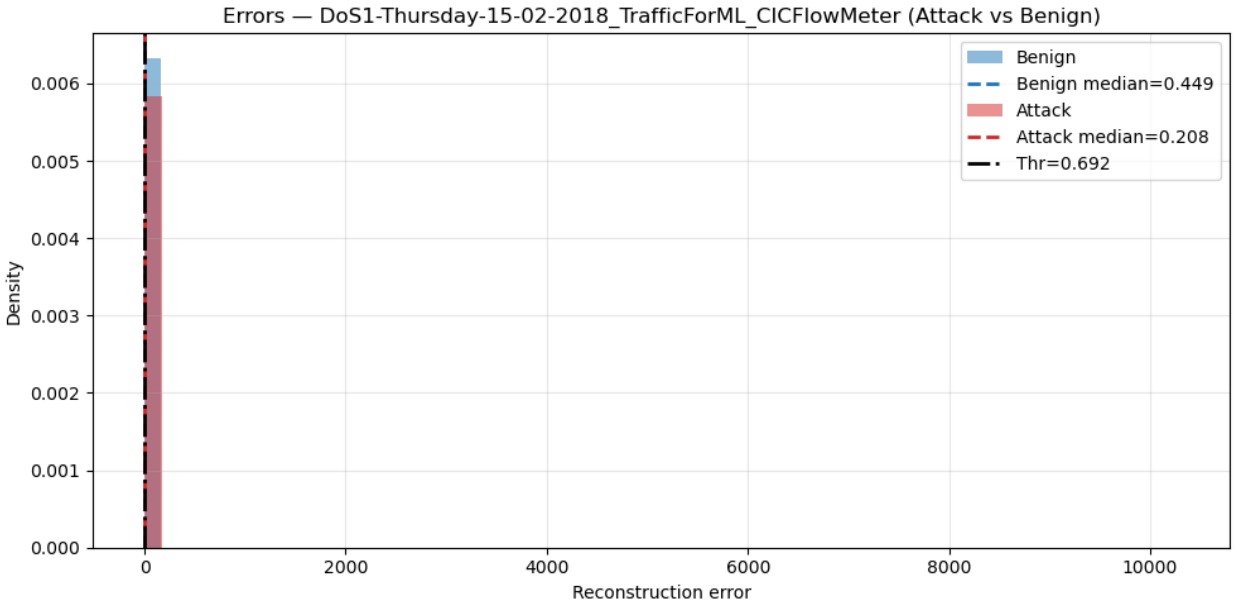


Figure20:ROC_DOS1-Thu

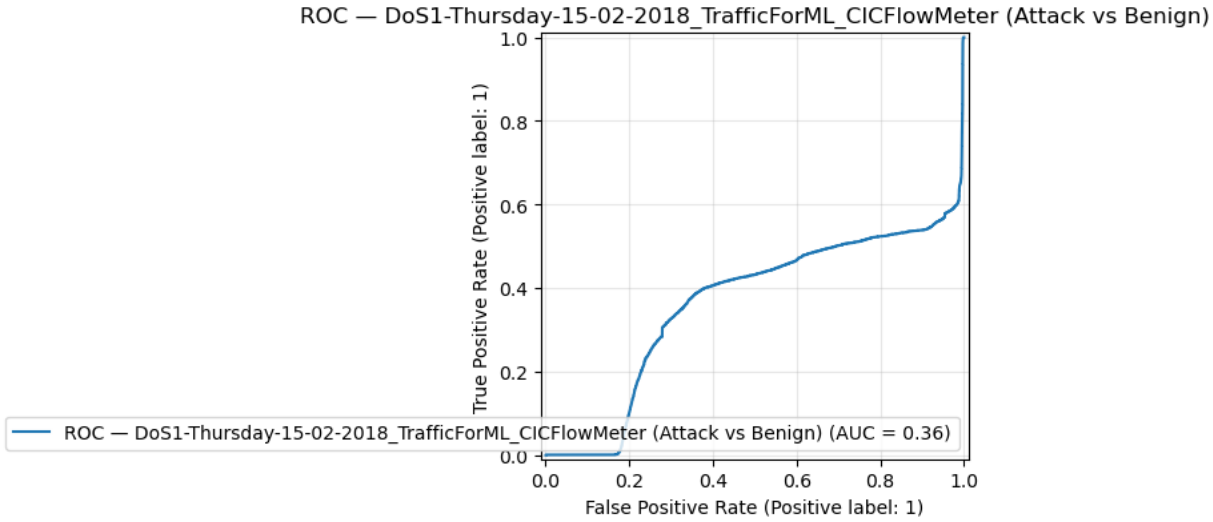
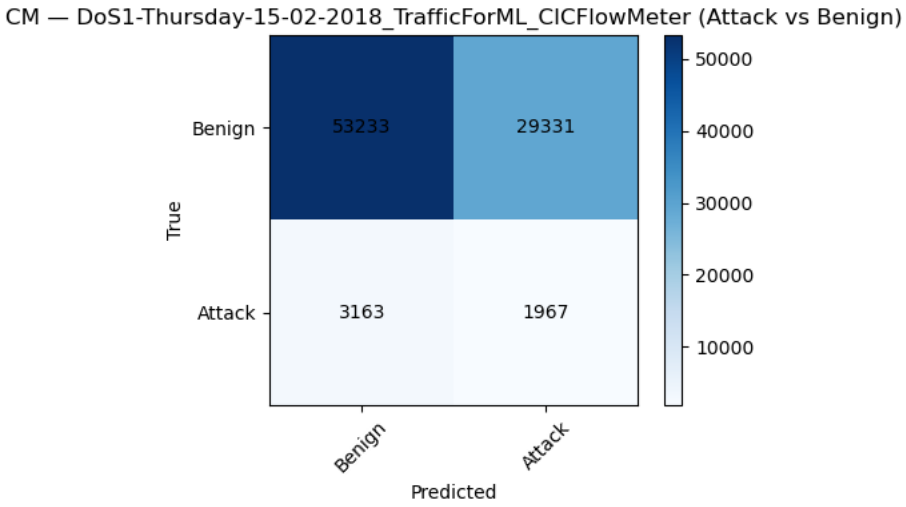


Figure21:CM_DOS1-Thu



Dos2-Friday-16-02-2018

Acc = 0.7542 | Prec = 0.0513 | Rec = 0.0001 | F1 = 0.0003 | ROC-AUC = 0.5521

thr_day_raw=0.046679 thr_used=0.692385

Interpretation: "Hard/failed" day — model almost completely blind to attacks despite decent accuracy.

The day is "tough": benign and attack scores overlap so much that the PR curve wants a very low threshold (0.0467) to get recall, which would massively increase benign false positives.

floor protects precision: the clamp prevents dropping below the global benign floor, so the final threshold is 0.6924. That's conservative; it will cut false positives but very likely drives recall near zero on this day

Figure22:RE_Error (Dos2-Fri)

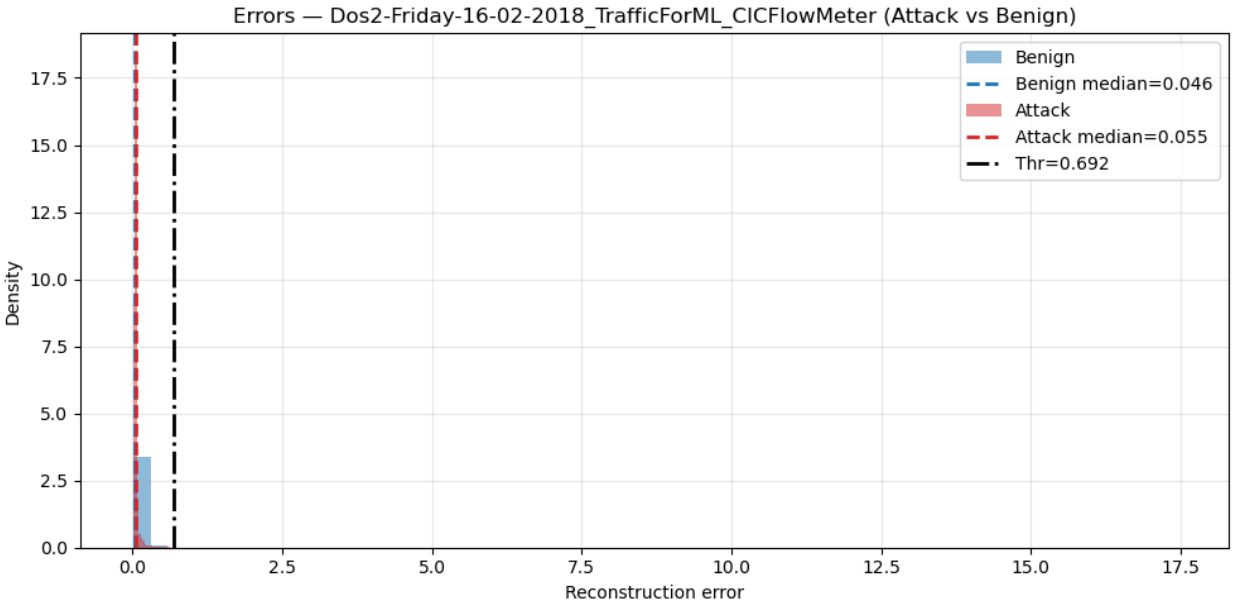


Figure22:ROC_Dos2-Fri

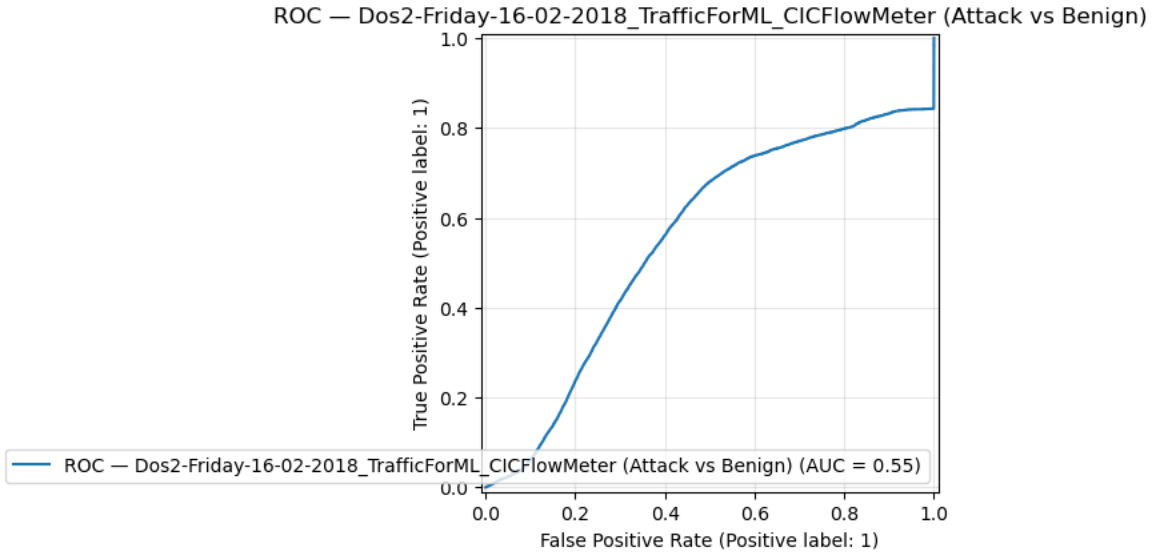
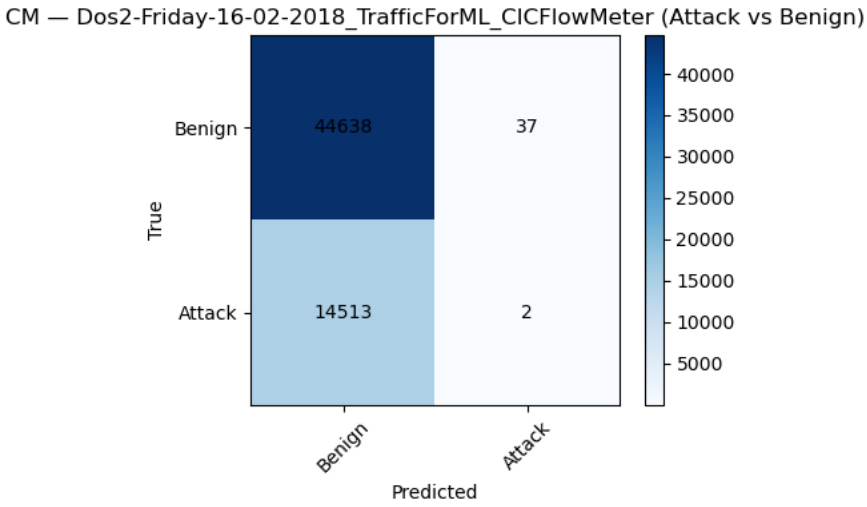


Figure23:CM_DOS2-Fri



Overall Insights

Good Generalization (2017 test days): AE performed well on DDoS/DoS traffic in CIC-IDS-2017, with high F1 (≈0.81–0.86) and ROC-AUC (0.71–0.92).

Degraded Generalization (2018 test days): On CIC-IDS-2018, performance became unstable:

Sometimes moderate (DDoS1 day),

Sometimes catastrophic (Dos2, DoS1 days).

Failure Modes:

Dos2 day → high false negatives (missed attacks).

DoS1 day → high false positives (benign flagged as attack).

Key Point: cross-dataset tests reveal **sensitivity to traffic distribution shifts**, with performance ranging from excellent to near-random.

Conclusion

This dissertation systematically evaluated the intra-dataset and cross-dataset generalization capabilities of machine learning models (Random Forest, XGBoost, Logistic Regression) and a deep learning model (LSTM Autoencoder) for intrusion detection. The results reveal a sharp contrast between in-dataset performance and cross-dataset robustness, underscoring the challenges of deploying IDS in real-world, dynamic environments.

Intra-Dataset Evaluation (Train & Test on Same Year)

Tree-based models (RF, XGBoost): Achieved near-perfect detection for major volumetric attacks (DoS/DDoS), with F1, Precision, and Recall consistently ~0.9–1.0. Even low-support attacks (e.g., Botnet, Infiltration) were detected provided they appeared in the training set. These results show that tree-based models capture attack signatures effectively when the training and test distributions match.

Logistic Regression (LR): Delivered moderate performance overall. Large-scale attacks (DoS/DDoS) were detected with Recall and F1 in the 0.8–0.95 range, but smaller or rare attacks suffered from poor F1-scores due to class imbalance and limited representation. LR benefits from linear separability in volumetric traffic features but struggles with complex or underrepresented attacks.

Cross-Dataset Evaluation (Train on 2018, Test on 2017, and vice versa)

ML models (RF, XGBoost, LR): Performance collapsed when models were trained on one dataset year and tested on the other. Recall frequently dropped to 0.0 for key attacks such as DDoS and Botnet, with DoS detection showing only limited generalization (recall 0.01–0.47 at best). Logistic Regression occasionally detected DDoS with recall ~0.88 on a

single 2018→2017 test day, but this success was not sustained across other days. Precision sometimes appeared high, but this was misleading since only a tiny fraction of true positives were detected. Overall, results confirm that ML models overfit to dataset-specific distributions and fail to generalize.

LSTM Autoencoder (AE): Demonstrated significantly better robustness in cross-dataset testing. Trained only on benign + DoS/DDoS traffic, the AE identified anomalies more consistently across datasets. Recall values for DoS/DDoS ranged between 0.4–0.7, far higher than tree-based or linear models under cross-testing. This highlights the strength of anomaly detection approaches, as the AE learns generic traffic reconstruction patterns rather than dataset-specific classification boundaries.

Future Work

This study highlights both the strengths and limitations of supervised ML models and LSTM Autoencoders for intra- and cross-dataset generalization. Based on these findings, several promising directions can be explored in future research:

- **Domain Adaptation & Transfer Learning:**
  Adaptation techniques such as adversarial training, feature alignment, or domain-invariant representation learning can be applied to mitigate dataset shift between different IDS benchmarks.
- **Transformer-Based Architectures:**
  Leveraging transformer models (e.g., BERT or time-series transformers) for IDS could capture long-range dependencies in traffic features and offer improved generalization across environments.
- **Hybrid IDS Frameworks:**
  Combining supervised ML with unsupervised anomaly detection (e.g., AE + XGBoost) could yield more resilient IDS solutions capable of detecting both frequent volumetric attacks (DoS/DDoS) and rare stealthy attacks.

- **Enhanced Rare Attack Handling:**

  Explore data augmentation strategies, synthetic traffic generation, or advanced rebalancing methods to better capture infiltration, botnet, and other low-support attacks.
- **Real-Time & Online Learning:**

  Incorporating incremental or online learning methods would allow IDS models to adapt continuously to evolving network behaviors and attacker strategies in real-world deployments.

REFERENCES

[1] Buczak, A.L.; Guven, E. "A survey of data mining and machine learning methods for cyber security intrusion detection." *IEEE Commun. Surv. Tutor.*, 2016, 18(2), 1153–1176.

[2] Liu, Y.; Liu, S.; Zhao, X. "Intrusion detection algorithm based on convolutional neural network." *DEStech Trans. Eng. Technol. Res.*, 2018, 37(12), 1271–1275. doi:10.12783/dtetr/iceta2017/19916.

[3] Srinivas, T.A.S.; Manivannan, S.S. "Prevention of hello flood attack in IoT using combination of deep learning with improved rider optimization algorithm." *Comput. Commun.*, 2020, 163, 162–175. doi:10.1016/j.comcom.2020.03.031.

[4] Malliga, S.; Nandhini, P.S.; Kogilavani, S.V. "A Comprehensive Review of Deep Learning Techniques for the Detection of (Distributed) Denial of Service Attacks." *Inf. Technol. Control*, 2022, 51, 180–215.

[5] Chimphlee, S.; Chimphlee, W. "Machine learning to improve the performance of anomaly-based network intrusion detection in big data." *Indones. J. Electr. Eng. Comput. Sci.*, 2023, 30, 1106–1119.

[6] A. Satyadas, U. Harigopal, 'Knowledge management tutorial: an editorial overview". *IEEE Transactions on System, Man and Cybernetics— Part C Application and reviews 31*, vol.31.pp.429-437, Nov 2001.

[7] Zeeshan Ali , Andrea Marotta AND Dajana Cassioli , *"* BERT Transformer Learning Approach and MLP for Intrusion Detection in Imbalanced Network Traffic*".*

[8] Joloudari, J.H.; Marefat, A.; Nematollahi, M.A.; Oyelere, S.S.; Hussain, S. "Effective class-imbalance learning based on SMOTE and convolutional neural networks." *Appl. Sci.*, 2023, 13(6), 4006.

[9] Zhang, H.; Huang, L.; Wu, C.Q.; Li, Z. "An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset." *Comput. Netw.*, 2020, 177, 107315.