

**TOWARDS ROBUST AND GENERALIZABLE
MACHINE LEARNING-BASED NETWORK
INTRUSION DETECTION: FROM ENSEMBLE
BASELINES TO DEEP LEARNING METHODS
FOR COMPLEX AND EVOLVING
THREAT ENVIRONMENTS**

**A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of**

MASTER OF TECHNOLOGY

in

Software Engineering

by

Suraj Prasad Kalauni

(24/SWE/03)

Under the Supervision of

Prof. Ruchika Malhotra

**Head of Department, Department of Software Engineering
Delhi Technological University, New Delhi**



Department of Software Engineering

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-110042, India

May, 2026

ACKNOWLEDGEMENTS

I would hereby like to express my heartfelt gratitude to **Prof. Ruchika Malhotra**, Head of Department, Department of Software Engineering, Delhi Technological University, for the support & guidance while compiling this report. Her insights and expertise were instrumental in shaping the direction of my research.

I would like to thank all the teachers and students of the Software Engineering department at Delhi Technological University for providing continuous support and valuable suggestions. Lastly, I would like to thank all my family and friends for providing continuous support and motivation during the process of this study.

Suraj Prasad Kalauni

(24/SWE/03)

Department of Software Engineering

Delhi Technological University

May, 2026



DELHI TECHNOLOGICAL UNIVERSITY

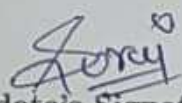
(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-42

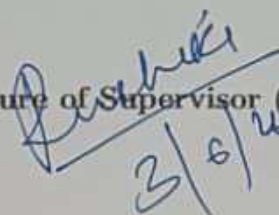
CANDIDATE'S DECLARATION

I Suraj Prasad Kalauni hereby certify that the work which is being presented in the thesis entitled TOWARDS ROBUST AND GENERALIZABLE MACHINE LEARNING-BASED NETWORK INTRUSION DETECTION: FROM ENSEMBLE BASELINES TO DEEP LEARNING METHODS FOR COMPLEX AND EVOLVING THREAT ENVIRONMENTS in partial fulfillment of the requirements for the award of the Degree of Master of Technology, submitted in the Department of Software Engineering, Delhi Technological University is an authentic record of my own work carried out during the period from 2024 to 2026 under the supervision of Prof. Ruchika Malhotra.

The matter presented in the thesis has not been submitted by me for the award of any other degree of this or any other Institute.


Candidate's Signature

This is to certify that the student has incorporated all the corrections suggested by the examiners in the thesis and the statement made by the candidate is correct to the best of our knowledge.


Signature of Supervisor (s)

Signature of External
Examiner



DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daultapur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR(s)

Certified that **Suraj Prasad Kalauni** (enrollment no. 24/SWE/03) has carried out their search work presented in this thesis entitled "**Towards Robust and Generalizable Machine Learning-Based Network Intrusion Detection: From Ensemble Baselines to Deep Learning Methods for Complex and Evolving Threat Environments**" for the award of **Master of Technology** from Department of Software Engineering, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature

Prof. Ruchika Malhotra

Head of Department

Department of Software Engineering

Delhi Technological University

Date: _____

3/6/26

Towards Robust and Generalizable Machine Learning-Based Network Intrusion Detection: From Ensemble Baselines to Deep Learning Methods for Complex and Evolving Threat Environments

Suraj Prasad Kalauni

ABSTRACT

Machine Learning Based NIDS (Network Intrusion Detection Systems) attain almost flawless performance on standardized benchmarks but there are three major problems with such systems which include sensitivity towards class imbalance, vulnerability to adversarial attacks, and lack of generalization to different network scenarios. This thesis addresses all three through a progressive three-stage research programme on the CICIDS2017, CSE-CICIDS2018, and TON IoT datasets.

In the first stage, a reproducible IDS pipeline is constructed on CICIDS2017 using imputation, standard scaling, SMOTE balancing, and stratified splitting. Random Forest, XGBoost, and MLP are trained and evaluated under identical conditions. XGBoost achieves the best performance (AUC = 0.995, FNR = 0.05%), and feature importance analysis identifies a compact top 10 feature subset accounting for 71.4% of total model importance.

In the second stage, the adversarial vulnerability of the baseline models is systematically evaluated. White box FGSM and PGD attacks on the MLP raise FNR from 0.30% to 33.58% at $\epsilon = 0.1$ a 112 fold increase in missed attacks. Black box transfer attacks on Random Forest and XGBoost reveal a novel FNR saturation phenomenon: both tree models reach approximately 34% FNR at the minimum perturbation budget $\epsilon = 0.01$ and show no further degradation with increasing perturbation, demonstrating that non differentiable classifiers are not inherently robust against transfer based evasion. Adversarial training recovers 15.4 percentage points of adversarial accuracy at a clean accuracy cost of only 2.0%.

In the third stage, three heterogeneous datasets are integrated through a unified preprocessing pipeline and a compact eight-feature shared representation. Four deep learning architectures, namely CNN, LSTM, GRU, and a lightweight Transformer, are evaluated on the merged two-million flow corpus. Under within-distribution evaluation, the GRU achieves the best result (94.23% accuracy, F1 = 0.8714), with all four architectures exceeding 91%, demonstrating that compact flow representations support reliable detection across heterogeneous network environments without dataset-specific feature engineering. A leave-one-dataset-out evaluation further reveals that cross-domain F1 scores range from 0.45 to 0.69 when models are tested on entirely unseen network environments, establishing a reproducible baseline and confirming that distribution shift between enterprise and IoT traffic remains a fundamental open challenge.

Keywords: Network Intrusion Detection, CICIDS2017, Random Forest, XGBoost, MLP, Adversarial Robustness, FGSM, PGD, Transfer Attack, Adversarial Training, False Negative Rate, Cross-Domain Generalisation, GRU, LSTM, Transformer.

LIST OF PUBLICATIONS

Publications arising from the research work presented in this thesis are listed below in chronological order.

Conference Papers

C1. Submission ID: 951

Title: “Network Intrusion Detection for Vulnerable Networks using CICIDS2017: Preprocessing, SMOTE Balancing, and Comparative Evaluation of Random Forest, XGBoost and Neural Network”

Conference: 9th International Conference on Innovative Computing and Communication (ICICC-2026)

Organized By: Shaheed Sukhdev College of Business Studies, University of Delhi, New Delhi

In association with National Institute of Technology Patna & University of Valladolid, Spain

Date: 06-07 February 2026

Authors: R. Malhotra and S. Prasad Kalauni

Status: [Accepted]

C2. Submission ID: 1532

Title: “Adversarial Evasion Attacks and Defense Strategies for ML-Based Network Intrusion Detection on CICIDS2017”

Conference: 7th International Conference on Data Analytics & Management (ICDAM-2026)

Organized By: London Metropolitan University (UK)

In association with WSG University, Portalegre Polytechnic University, SGGW Management Institute

Date: 12-14 June 2026

Authors: S. Prasad Kalauni and R. Malhotra

Status: [Accepted]

C3. Submission ID: 1585

Title: “A Cross-Domain Intrusion Detection System Using Unified Network Flow Representations and Deep Models”

Conference: 7th International Conference on Data Analytics & Management (ICDAM-2026)

Organized By: London Metropolitan University (UK)

In association with WSG University, Portalegre Polytechnic University, SGGW Management Institute

Date: 12-14 June 2026

Authors: S. Prasad Kalauni and R. Malhotra

Status: [Accepted]

Contents

Acknowledgements	ii
Candidate's Declaration	iii
Certificate by the Supervisor	iv
Abstract	v
List of Publications	vi
List of Tables	xi
List of Figures	xii
List of Symbols, Abbreviations, and Nomenclature	xiii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Network Intrusion Detection: An Overview	2
1.2.1 Challenges That Motivated This Work	2
1.3 Research Motivation and Problem Framing	3
1.4 Contributions of This Thesis	3
1.5 Thesis Organisation	4
2 Literature Review	5
2.1 Introduction	5
2.2 Machine Learning for Network Intrusion Detection	5
2.2.1 From Signatures to Learned Representations	5
2.2.2 Ensemble Methods and Gradient Boosting	5
2.2.3 Preprocessing and Class Imbalance	6
2.2.4 Deep Learning Approaches	6
2.3 Recurrent and Transformer Architectures for IDS	6
2.3.1 LSTM and GRU Networks	6
2.3.2 Transformer Architectures	7
2.4 Cross-Domain Generalisation in Intrusion Detection	7
2.4.1 The Single-Dataset Problem	7
2.4.2 Feature Alignment and Multi-Dataset Training	7
2.4.3 Federated Learning as an Alternative	8
2.5 Adversarial Attacks on ML-Based NIDS	8
2.5.1 The Foundations	8
2.5.2 Transferability	8
2.5.3 Adversarial Attacks in the NIDS Domain	8

2.6	Defence Mechanisms	9
2.6.1	Adversarial Training	9
2.6.2	Input Preprocessing and Certified Robustness	9
2.7	Benchmark Datasets	10
2.7.1	Historical Context	10
2.7.2	CICIDS2017	10
2.7.3	CSE-CICIDS2018	10
2.7.4	TON IoT	10
2.8	Research Gaps This Thesis Addresses	10
3	Problem Statement and Research Objectives	12
3.1	Motivation	12
3.2	Problem Formulation	12
3.2.1	Stage 1: Building a Trustworthy Baseline	12
3.2.2	Stage 2: Probing Adversarial Vulnerability	13
3.2.3	Stage 3: Cross-Domain Generalisation	13
3.3	Research Questions	13
4	Methodology	15
4.1	Overview	15
4.2	Data Preprocessing	15
4.2.1	CICIDS2017 Pipeline (Stages 1 and 2)	15
4.2.2	Multi-Dataset Pipeline (Stage 3)	16
4.3	Stage 1: Ensemble and Neural Network Classifiers	16
4.4	Stage 3: Deep Learning Architectures	17
4.5	Stage 2: Adversarial Attacks and Defence	18
4.5.1	Threat Model	18
4.5.2	Attack Algorithms	18
4.5.3	Adversarial Training Defence	18
4.6	Evaluation Metrics	18
5	Proposed System and Model Design	20
5.1	System Architecture Overview	20
5.2	Zone 1a: Single-Dataset Pipeline (Stages 1 and 2)	22
5.2.1	Data Ingestion	22
5.2.2	Cleaning and Preparation	22
5.2.3	Scaling and Balancing	22
5.3	Zone 1b: Multi-Dataset Pipeline (Stage 3)	22
5.3.1	Dataset Integration	22
5.3.2	Feature Alignment	22
5.3.3	Scaling and Class Weights	23
5.4	Zone 2a: Ensemble and MLP Model Bank (Stages 1 and 2)	23
5.4.1	Random Forest Configuration	23
5.4.2	XGBoost Configuration	23
5.4.3	MLP Architecture	24
5.4.4	Feature Importance	24
5.5	Zone 2b: Deep Learning Model Bank (Stage 3)	24

5.6	Zones 3–5: Adversarial Evaluation and Defence (Stage 2)	25
5.6.1	Attack Generation (Zone 3)	25
5.6.2	Evaluation Engine (Zone 4)	25
5.6.3	Defence Module (Zone 5)	25
6	Experimental Setup	26
6.1	Overview	26
6.2	Datasets	26
6.2.1	CICIDS2017 (Stages 1 and 2)	26
6.2.2	CSE-CICIDS2018 (Stage 3)	27
6.2.3	TON IoT (Stage 3)	27
6.2.4	Unified Corpus (Stage 3)	28
6.3	Compute Environment	28
6.4	Model Training Details	28
6.4.1	Stage 1: Ensemble Classifiers and MLP	28
6.4.2	Stage 2: Adversarial Training Defence	29
6.4.3	Stage 3: Deep Learning on Unified Corpus	29
6.5	Evaluation Protocols	29
6.5.1	Stage 1	29
6.5.2	Stage 2	29
6.5.3	Stage 3	30
6.6	Reproducibility	30
7	Results and Discussion	31
7.1	Overview	31
7.2	Stage 1: Baseline Performance on CICIDS2017	31
7.2.1	Quantitative Results	31
7.2.2	ROC Analysis	31
7.2.3	Confusion Matrix Analysis	32
7.2.4	Feature Importance	33
7.2.5	Stage 1 Summary	33
7.3	Stage 2: Adversarial Robustness Evaluation	33
7.3.1	Attack Impact at $\epsilon = 0.1$	33
7.3.2	FNR Saturation in Tree-Based Models	35
7.3.3	Adversarial Training Results	36
7.3.4	Stage 2 Summary	37
7.4	Stage 3: Cross-Domain Deep Learning	37
7.4.1	Training Convergence	37
7.4.2	Performance Comparison	37
7.4.3	Cross-Domain Generalization Evaluation	38
7.4.4	Why the GRU Performs Best	38
7.4.5	Evaluation Curves	39
7.4.6	Stage 3 Summary	40
7.5	Cross-Cutting Discussion	40
7.5.1	Answers to Research Questions	40
7.5.2	Connecting the Three Stages	41

7.5.3 Broader Observations	41
8 Conclusion, Future Scope and Social Impact	42
8.1 Conclusion	42
8.2 Future Scope	43
8.3 Social Impact	44
References	45
Appendices	48
A Additional Experimental Results	49
Publication Proof Section	50
1. List of Publications	50
2. Paper Acceptance Proofs	51
2.1 C1: ICICC 2026 Acceptance Letter	51
2.2 C2: ICDAM 2026 Acceptance Letter (Paper ID 1532)	52
2.3 C3: ICDAM 2026 Acceptance Letter (Paper ID 1585)	53
3. Indexing Proof	54
3.1 ICICC 2026: Springer LNNS Indexing Proof	54
3.2 ICDAM 2026: Springer LNNS Indexing Proof	55
4. Conference Certificate	56
4.1 ICICC 2026: Presenter Certificate	56
5. Conference Paper Registration Receipts	57
5.1 ICICC 2026: Registration Receipt	57
5.2 ICDAM 2026: Registration Receipt	58
Author's Profile	59

List of Tables

5.1	System Components and Their Role Across Research Stages	20
5.2	Shared Feature Set for Cross-Domain Framework	23
5.3	Random Forest Hyperparameters	23
5.4	XGBoost Hyperparameters	23
5.5	MLP Architecture	24
5.6	Shared Deep Learning Training Configuration	24
6.1	CICIDS2017 Dataset Statistics	26
6.2	Attack Category Distribution in CICIDS2017 (Sampled Subset)	27
6.3	Unified Multi-Dataset Corpus Statistics	28
6.4	Compute Environment	28
6.5	Adversarial Training Configuration	29
6.6	Deep Learning Training Configuration (Stage 3)	29
7.1	Baseline Model Performance on CICIDS2017 Test Partition	31
7.2	Top-10 Features by RF Importance	33
7.3	Model Performance Under Adversarial Attack ($\epsilon = 0.1$)	34
7.4	FNR Values Across Models, Attack Types, and Perturbation Magnitudes	36
7.5	Defence Effectiveness: Undefended vs Defended MLP	36
7.6	Deep Learning Architecture Comparison on Unified Corpus	37
7.7	Cross-Domain Evaluation Results Using GRU (Leave-One-Dataset-Out Protocol)	38
A.1	Table-I.1: MLP Per-Class Metrics — Clean vs. FGSM ($\epsilon = 0.1$)	49
A.2	Table-I.2: Accuracy vs. ϵ for All Models Under FGSM	49

List of Figures

5.1	End to end adversarial robustness evaluation pipeline	21
5.2	Unified preprocessing pipeline for cross domain dataset integration . .	21
7.1	ROC curves for RF, XGBoost, and MLP on the CICIDS2017 test partition. AUC values are annotated in the legend.	32
7.2	Confusion matrices for RF, XGBoost, and MLP on the CICIDS2017 test partition.	32
7.3	Confusion matrices for the MLP on unperturbed (left) and FGSM-perturbed (right) test sets for $\varepsilon = 0.1$. The number of false negatives increases from 69 to 7,605.	34
7.4	Accuracy under clean, FGSM, and PGD conditions for all three models at $\varepsilon = 0.1$	35
7.5	FNR versus perturbation magnitude ε under FGSM. Tree-based models saturate immediately at $\varepsilon = 0.01$; MLP FNR grows monotonically. . .	35
7.6	ROC curve for the Transformer on the unified test partition (AUC = 0.924). .	39
7.7	Precision-recall curve for the Transformer on the unified test partition. .	40

LIST OF SYMBOLS, ABBREVIATIONS, AND NOMENCLATURE

Abbreviations

AUC	Area Under the Receiver Operating Characteristic Curve
CICIDS	Canadian Institute for Cybersecurity Intrusion Detection Evaluation
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CSV	Comma-Separated Values
DDoS	Distributed Denial of Service
DTU	Delhi Technological University
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
FGSM	Fast Gradient Sign Method
GPU	Graphics Processing Unit
IDS	Intrusion Detection System
IoT	Internet of Things
LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multilayer Perceptron
NIDS	Network Intrusion Detection System
PGD	Projected Gradient Descent
RF	Random Forest
ROC	Receiver Operating Characteristic
SMOTE	Synthetic Minority Over-sampling Technique
TF	TensorFlow
TN	True Negative
TP	True Positive
VRAM	Video Random Access Memory
XGBoost	Extreme Gradient Boosting

Mathematical Symbols and Notation

$\mathbf{x}$	Clean input feature vector
$\mathbf{x}_{\text{adv}}$	Adversarially perturbed input feature vector
δ	Adversarial perturbation vector
ε	Perturbation budget (L_∞ norm bound)
α	Step size in PGD iterations
K	Number of PGD iterations
ρ	Adversarial mixing ratio in adversarial training
θ	Model parameters (weights)
$\mathcal{L}$	Loss function (binary cross-entropy)
y	True class label
$\hat{y}$	Predicted class label
Π	Projection operator onto ε -ball
$\mathcal{S}$	ε -ball constraint set
$\nabla_{\mathbf{x}} \mathcal{L}$	Gradient of loss w.r.t. input
$\ \cdot\ _\infty$	L_∞ (Chebyshev / max) norm
$\mathbb{E}$	Expectation operator
$\mathcal{D}$	Data distribution

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Intrusion Detection Systems play an important role in this case by monitoring network traffic and issuing notifications whenever any suspicious activity occurs. For many years, such IDSs used an exhaustive database of attack signatures which had to be manually updated whenever a new type of attacks emerged. Such approach worked well with already known types of attacks but failed when dealing with any other types.

The move towards machine learning approaches made everything different. As opposed to previous IDSs, these approaches don't rely on comparison with rules and rather learn to recognize patterns of benign and malicious traffic using labeled data as an example and then use this knowledge in real-life situations [1, 2]. Flow-level classifiers trained using benchmarks, such as CICIDS2017, became the yardstick for such IDSs, yielding quite amazing results in terms of AUC value, which is often greater than 0.99 (almost perfect result) [3–5].

Nevertheless, there remains a difference between the two. This difference sparked this thesis.

There are three common issues that have been encountered when looking into IDS research in depth. First, there is the problem of class imbalance. It is clear that network flows are mostly benign, whereas attacks are only a small proportion of overall traffic and may be very small. A model that labels all network traffic as benign would have relatively high accuracy, but it would not detect any attacks. Class imbalance is thus a silent distortion in evaluations that needs to be handled properly.

The second issue is related to the adversarial vulnerability of classifiers. The AUC scores obtained for test networks in the case of CICIDS2017 are based on an assumption that test traffic is generated by the same attacker as the training network. This means that the adversary who knows about the use of the particular IDS may craft traffic in such a way that it appears benign for the classifier. This creates a vulnerability that cannot be revealed through benchmarks alone. In practice, 99% accurate classifiers may miss one-third of attacks [6].

Thirdly, generalization to other environments is also a concern. Many studies on IDS train and test their methods using only a single dataset that is collected in a particular environment. An IDS trained using enterprise network data from 2017 would be very different in its performance when used for detecting attacks in a cloud environment or Internet of Things (IoT) environment. Very few studies have actually focused on testing the applicability of the pattern found using one dataset to other datasets, and fewer have actually tried building IDSs designed to generalize.

In this paper, our aim is to tackle all three problems one by one, understanding their interconnectedness as opposed to treating them as individual problems. Firstly, we set up a base which is replicable and works well with CICIDS2017 data, taking into consideration any challenges involved in preprocessing. In the second step, we examine what happens when the adversary attempts to circumvent these models, systematically

measuring performance decline and testing whether adversarial training can improve robustness. The third stage, taking a step back from the single dataset issue, proposes the creation of a common training framework for all datasets, i.e., CICIDS2017, CSE-CICIDS2018, and TON IoT.

1.2 Network Intrusion Detection: An Overview

In the case of an anomaly detection IDS, it learns how normal traffic should look while training. After that, it reports all those events that are different from the normal traffic. Flow-level input features are used as the inputs in these IDSs. They refer to the statistics generated for each TCP or UDP connection. These statistics include packets counts, bytes, time duration of the flow, inter arrival times, and many more. A total of 79 features per flow are provided by CICIDS2017 dataset. This number of features might be considered enough to build good classifiers, but it may cause issues with irrelevant or related features.

DDoS attack, PortScan attacks, different web application attacks (such as SQLi or XSS attacks), and Infiltration make up the attacks in CICIDS2017 dataset.

1.2.1 Challenges That Motivated This Work

Three issues come up almost immediately when working with CICIDS2017 and thinking about applying it.

1. Class imbalance occurs at a structural level, in the nature of the data itself. In CICIDS2017, even after resampling, the benign to attack ratio may still go beyond 4:1. Not dealing with this leads to classifiers biased towards predicting the majority class. This means low recall rates on attack flows, which is what we care most about here. SMOTE can mitigate it, though it should be applied cautiously to avoid leakage of information from training to evaluation.
2. The second issue, adversarial evasion, is harder to spot. The point is that if an attacker has studied or cloned our IDS classifier, he/she can modify attack flows in ways that would force the classifier to predict benign flow instead. It becomes an important issue for network security because failure to detect attacks is much more harmful than false positives. For this reason, the important metric here is False Negative Rate rather than accuracy, which was mentioned above. Note that it is not exclusive to neural networks, and tree-based classifiers (Random Forest and XGBoost) may become vulnerable due to transferability.
3. Cross-domain generalization is the question of whether an IDS model learned in one environment may be relied upon in another. Very few studies of IDS tackle this problem by testing against held-out data drawn from the same capture as the data used to train it; in practice, this asks a very different question. To design systems capable of generalizing across environments, there is a need to rethink both feature representation and learning procedures.

1.3 Research Motivation and Problem Framing

All three challenges are interrelated in that one needs to set up a baseline first before studying adversarial robustness because degradation of adversarial robustness is relative. Moreover, the feature importance result of the baseline challenge informed our choice of flow-level features in the cross-domain generalization challenge. Hence, all three stages of work were consciously linked in some way.

The research questions this thesis addresses are:

- **RQ1 (Baseline):** Having preprocessed and balanced CICIDS2017 dataset, which classifier among Random Forest, XGBoost, and MLP performed better at detecting attacks? What features are the most discriminative?
- **RQ2 (Robustness):** To what extent are FGSM and PGD attacks harmful to MLP, and how well can transfer attacks harm RF and XGBoost? Is there a noticeable difference in how gradient-based and tree-based models react to increasing levels of perturbation? Can adversarial training restore meaningful robustness without greatly sacrificing clean accuracy?
- **RQ3 (Generalisation):** Can a deep learning framework trained on a merged corpus from three heterogeneous datasets — using only a compact shared feature set — achieve reliable binary detection across enterprise, extended enterprise, and IoT traffic environments?

1.4 Contributions of This Thesis

The work presented in this thesis makes the following contributions:

1. **A reproducible IDS baseline on CICIDS2017.** A complete preprocessing and training pipeline covering imputation, standard scaling, SMOTE oversampling, and stratified splitting is developed and evaluated across Random Forest, XGBoost, and MLP. The three models achieve AUC values of 0.992, 0.995, and 0.994 respectively. Feature importance analysis identifies the most informative flow attributes and shows that a compact top-10 subset accounts for the bulk of model discriminative power.
2. **A unified adversarial evaluation framework.** FGSM and PGD white-box attacks on the MLP demonstrate a 112-fold increase in missed attacks at $\epsilon = 0.1$. Transfer attacks on RF and XGBoost reveal an immediate FNR saturation near 34% at $\epsilon = 0.01$, which persists regardless of how much the perturbation budget is increased — a finding that has not been documented in prior work and has direct implications for how practitioners think about tree-based classifier security.
3. **Adversarial training as a practical defence.** Single-model adversarial retraining recovers 15.4 percentage points of adversarial accuracy at a clean-accuracy cost below 2%, which is a trade-off that production deployments can reasonably accept. This demonstrates that meaningful robustness is achievable without the complexity of ensemble-level defences.

4. **A cross-domain deep learning framework.** Integrating CICIDS2017, CSE-CICIDS2018, and TON_IoT through a unified preprocessing pipeline and an eight-feature shared representation, four deep learning architectures are trained and compared on approximately two million flow records. The GRU achieves 94.23% accuracy with $F1 = 0.8714$ under within-distribution evaluation, and all four architectures exceed 91%. A leave-one-dataset-out evaluation establishes cross-domain F1 scores of 0.45 to 0.69, quantifying the distribution gap between enterprise and IoT network environments and providing a reproducible baseline for future domain adaptation work.

1.5 Thesis Organisation

- **Chapter 2** literature on ML-based NIDS, adversarial attacks, defense strategies, and cross-domain generalization. It specifies the gaps that this thesis fills.
- **Chapter 3** defines the research problem and presents the three research questions.
- **Chapter 4** describes the approach used at each of the three phases. This includes the preprocessing pipeline, attack algorithms, defensive algorithm, and metrics.
- **Chapter 5** illustrates the system architecture and model designs of each phase.
- **Chapter 6** describes the experiments setup. It tells of features of your data set, how the model has been trained, which computers used
- **Chapter 7** reports and talks about findings on the experimental data in all 3 phases.
- **Chapter 8** concludes by providing a summary of the results, areas for future investigation, and the relevant social implications.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter reviews the the relevant literature and discusses the state-of-the-art works performed in the field of the three research themes of this thesis. The literature review is structured in: ML Techniques for network intrusion detection (focused on preprocessing and classification/selection model) Adversarial attacks to ML based models and the defense against them Cross-domain generalization & Multi-dataset training benchmark. The chapter concludes by highlighting the specific gaps that inspired the research contributions discussed in later chapters.

2.2 Machine Learning for Network Intrusion Detection

2.2.1 From Signatures to Learned Representations

The earliest IDS approaches depended on signature databases, which were manually maintained collections of known attack patterns network traffic was matched against. These systems worked well against well-documented threats but were reactive by nature. Anything outside the signature library went undetected. As attacks became more varied, the limitations of this method became clear. The scientific community began investigating approaches based on statistical and machine learning methods that could learn patterns for detecting attacks from the data rather than relying on expert-generated rules [1, 2].

This approach introduced capabilities as well as complexities. Machine Learning based detectors can theoretically detect new attack variations. To achieve this potential, these detectors need adequate training data preparation, proper feature selection, and proper evaluation methods.

2.2.2 Ensemble Methods and Gradient Boosting

Among the numerous model families considered by researchers in studies using flow-based IDS datasets, gradient boosting decision trees have proven themselves as most effective. XGBoost, LightGBM, and CatBoost often surpass neural networks on structured tabular data without much hyperparameter tweaking. Besides, they output feature importance values useful for interpreting models. Srivastav *et al.* [7] showed that ensemble techniques are superior to individual classification models when dealing with skewed data distributions. In addition, Rana *et al.* [8] proposed a framework with an enhanced minority recall capability. Varaprasad *et al.* [9] made comparisons between ML and DL algorithms and came up with the conclusion that neither method beats the other one.

Ahmad *et al.* [10] reached a similar conclusion. As noted in their analysis, decisions related to data preprocessing were more important than the model design itself when it came to affecting performance. The fact is often overlooked in scientific articles which focus more on designing new models than pre-processing data.

2.2.3 Preprocessing and Class Imbalance

Class imbalance is not a minor detail in IDS research but rather an intrinsic characteristic of the task. Normal network traffic dominates in volume, whereas attacks constitute only a minor part of records in datasets. Models trained on unbalanced data without any adjustments will likely focus on the majority class and exhibit bad recall of attacks. Multiple studies identified this problem as the main factor affecting IDS accuracy, outweighing the effect of the choice of algorithm on results [7, 8].

One of the methods used to tackle the problem of class imbalance that has proven its worth is the SMOTE technique. It creates synthetic samples of minority class via interpolating other samples from the same class. Shenbagalakshmi *et al.* [11] demonstrated that the use of SMOTE combined with feature selection can enable achieving high IDS accuracy – over 99% for certain types of attacks within CICIDS2017. The common approach suggested by multiple sources is to use SMOTE exclusively on training sets, while the test set remains unchanged. Otherwise, the performance metrics will be biased because of artificially removed class imbalance problems [3, 7, 8].

2.2.4 Deep Learning Approaches

Progress has been observed in the area of deep learning within IDS investigations, such as enriched inputs and packet level sequences. Getman *et al.* [12] proved that CNNs and dense models could work well on the dataset CICIDS2017. Farhan *et al.* [13] obtained a result above 99% of accuracy using a deep learning pipeline. Zhang *et al.* [4] reviewed the state-of-the-art works and found out that there were two primary obstacles in the field, namely spatiotemporal feature extraction and imbalanced classification problems. Pinto Neto *et al.* [5] considered applications in recent technological conditions of IoT and 5G, where computational restrictions also added another problem to the existing issues.

Hybrid structures composed of CNNs and recurrent layers are of particular interest. Dai *et al.* [14] designed a CNN-BiLSTM-Attention model and indicated the benefits from spatial-temporal feature extraction in their paper. Chinnasamy *et al.* [15] performed a comprehensive investigation in the field and noted that the recurrent models were usually superior to feedforward ones in processing sequential traffic flow.

2.3 Recurrent and Transformer Architectures for IDS

2.3.1 LSTM and GRU Networks

Several papers applied LSTM and GRU networks into IDS based on the assumption that the network traffic has sequential characteristics – packets in one stream follow each other, and relationships between streams in sequence are indicative about attacks. Parakkal *et al.* [16] compared various deep learning models for NIDS, where recurrent models demonstrated better performance on the sequential data of flows than feedforward neural networks did. The work of Mebawondu and Ajisafe-Badeji [17] proved that combining recurrent layers in an ensemble configuration was more efficient in attack detection from diverse attack categories.

GRU architecture is a simpler modification of LSTM because it combines the forget and input gates into a single update gate, reducing the parameter count while

maintaining comparable modelling capacity. For applications where computational efficiency matters — and real-time network monitoring is one such application — this is a meaningful advantage.

2.3.2 Transformer Architectures

More recently, transformer models have been brought into IDS research. The mechanism used called self-attention allows the model to capture relationships between all pairs of positions in the input simultaneously, which is appealing when the relationships between different flow features are complex and non-local. Nemiwal *et al.* [18] evaluated transformer-based NIDS across multiple datasets and reported competitive detection rates. Akuthota and Bhargava [19] developed a transformer IDS specifically for IoT networks and showed that self-attention could effectively model the irregular traffic patterns in that domain.

That said, transformers are not universally better. For low dimensional tabular inputs such as the eight feature description from the Stage 3 in this thesis, the self attention mechanism may not contain enough structure for it to take full advantage of the model. Also reported in multiple publications is the finding that class-weighted optimisation can be crucial to successfully train stable transformer on imbalanced data distributions [18], an observation that is consistent with our empirical results.

2.4 Cross-Domain Generalisation in Intrusion Detection

2.4.1 The Single-Dataset Problem

This allows you to get some sense of how well the machine learned the characteristics of a specific network environment at a certain point in time however, this approach is not effective when assessing the performance of deployment in a new environment. The above drawback was emphasized by Cantone *et al.* [20], who observed that there is considerable deterioration in performance when classifiers are trained and tested using datasets belonging to two distinct classes. Extending the above observation, Kostas *et al.* [21] observed that the selection of packet features induces a dataset bias, which produces optimistic results for identical datasets.

What this means is that you cannot simply make the assumption that a model which has been trained using CICIDS2017 data can be efficient for IoT traffic analysis or any other enterprise network setting.

2.4.2 Feature Alignment and Multi-Dataset Training

One of the ways to overcome this limitation is to use multiple datasets at once during training, forcing the model to adapt to various traffic distributions. Kurnala *et al.* [22] proposed a cross-domain cybersecurity framework integrating multiple network environments and reported improved robustness. Aldaej *et al.* [23] demonstrated that exposure to diverse training sources reduced the overfitting seen with single-dataset models in IoT-edge IDS settings.

The practical challenge is that different datasets use different feature sets, collected by different tools in different environments. Direct model transfer is usually impossible without a harmonisation step. Westphal *et al.* [24] studied feature selection for cross-network IDS and found that flow-level aggregate statistics provide a more stable and

transferable representation than packet-level attributes. Kostas *et al.* [21] made a similar point, arguing that high-dimensional packet features create dataset-specific signatures that hurt generalisation.

2.4.3 Federated Learning as an Alternative

Rather than centralising data from multiple sources, federated learning allows organisations to collaboratively train a shared model without sharing raw traffic data. Hernandez-Ramos *et al.* [25] examined the technique thoroughly, and the main issues were found to be the heterogeneity of data distributions on the client machines, the communication costs involved, and the potential for poisoning attacks by malicious clients. Agrawal *et al.* [26] previously addressed the same problems. Though federated learning offers much promise, it has its own issues and is unrelated to the proposed cross-domain learning method.

2.5 Adversarial Attacks on ML-Based NIDS

2.5.1 The Foundations

The investigation of adversarial examples in machine learning research started when Goodfellow *et al.* [6] came up with Fast Gradient Sign Method and proved that the addition of carefully selected perturbations to an input would make deep neural networks classify it incorrectly. However, what made this discovery stand out was not simply the proof of the possibility of misclassifying; instead, it was the conclusion that the vulnerability resulted from linear property of models in high dimensional spaces, rather than an error that needed fixing.

Madry *et al.* [27] further developed the area with their invention of Projected Gradient Descent as a more effective attack, as well as with their consideration of adversarial robustness as a minimax optimization problem. Their approach of adversarial training, involving alternating between clean and perturbed inputs while training, became the gold standard of defense to compare any other solution against.

2.5.2 Transferability

Papernot *et al.* [28] analyzed what occurred when adversarial examples crafted for one model were used against another model—the so-called transfer attacks. It turned out that such adversarial examples were usually still able to cause a misclassification despite using different architectures for the source and target models, which implies serious real-world consequences of their creation. No access to the deployed model is needed to perform an evasion attack; all it takes is training a local model as a surrogate.

2.5.3 Adversarial Attacks in the NIDS Domain

The use of adversarial attacks on network intrusion detection systems became increasingly popular. In particular, He *et al.* [29] proposed a classification scheme for the types of adversarial threats to network intrusion detection systems, dividing evasion attacks on one hand from poisoning attacks on the other, depending on whether the adversary targeted the inference process or the training process, and stressing the ease of creating L_∞ -bounded perturbation for CICIDS2017 due to its continuous flow-level features. Alotaibi and Rassam [30] surveyed the state of the art from a practical point of view,

determining which attack schemes posed the greatest threat.

In particular, Catillo *et al.* [31] carried out a case study for CICIDS2017 and demonstrated that gradient-based attacks could effectively be transferred through tree-ensemble model boundaries, a finding directly applicable to our own experiment in this thesis.

One significant omission from many of the studies above is the emphasis on the measure of accuracy. In the setting of a security system, missing an attack is more costly than an alert, and hence the False Negative Rate is the relevant performance measure when evaluating defenses against adversaries. Papers generally do not conduct a sweeping experiment to find out the FNR across various levels of perturbation and hence, the real-world impact of the attack.

2.6 Defence Mechanisms

2.6.1 Adversarial Training

Adversarial training [27] is the only defence approach that has been empirically validated in many different contexts. The idea is to create adversarial examples on the fly at the time of training and add those to the training batch, teaching the model to learn robust boundaries from the examples. However, one disadvantage of this approach is the heavy computational requirements involved since generating adversarial examples is computationally expensive and thus the training takes twice or thrice as long.

Awad *et al.* [32] applied adversarial training along with label smoothing, noise addition and denoising autoencoders to an IDS setting for an ensemble method. Their experiments on the CICIDS2017 and CICIDS2018 datasets showed that the layered defense was better compared to all standalone defenses. Halvorsen *et al.* [33] reviewed techniques used in IDS using generative models and included adversarial data augmentation in the list of defense subcategories.

2.6.2 Input Preprocessing and Certified Robustness

Defenses based on input preprocessing aim at detecting or filtering out adversarial perturbations before the input arrives at the classifier. These include feature squeezing, random smoothing, and denoising autoencoder defenses, among others. While such defenses can be used to further secure classifiers trained via adversarial training, they may harm accuracy on clean samples and may be vulnerable to attacks that adapt to the specific defense mechanism.

Defense mechanisms offering formal robustness guarantees consist of demonstrating that the output of the classification model will be invariant to any input that lies within the radius of an L_p -ball defined around the original sample. Randomized smoothing is currently the only practically viable method for generating certificates, but it is only possible to generate certificates in the L_2 norm, and the certified radius is insufficiently large for IDS tests.

2.7 Benchmark Datasets

2.7.1 Historical Context

The KDDCup99 dataset has been for many years the gold standard IDS dataset, and comes from the DARPA 1998 simulation testbed. It has severe flaws by current standards, in that the data is unrealistic in its network traffic, temporally redundant, and uses attack classifications which are no longer relevant to current organizational threats. NSL-KDD improved on some of these flaws, yet is still quite limited.

2.7.2 CICIDS2017

CICIDS2017 [34] dataset, generated from Canadian Institute of Cybersecurity, on the other hand, is a significantly more realistic benchmark. This contains data related to network traffic over seven days in a lab environment where attacks actually take place, generating some 2.8 million flow logs based on 79 features through CICFlowMeter. The various types of attacks in this dataset include DDoS, PortScan, Web Attacks, Botnet, Infiltration, and Heartbleed. Due to its size, complexity, and variety, CICIDS2017 is the most popular benchmark for IDS evaluation and serves as the core dataset for Stages 1 and 2 of this thesis project.

2.7.3 CSE-CICIDS2018

As for the approach employed, CSE-CICIDS2018 uses the same principle as CICIDS2017, except that this time around there is an extended data set that covers ten days with more types of attacks. The flow extraction process is similar to that used in CICIDS2017 as a result, the datasets are fully compatible with one another.

2.7.4 TON IoT

The TON IoT dataset was created specifically for scenarios of IoT and IIoT, and the traffic of this dataset has distinct features compared to other enterprise network datasets. The dataset includes seven kinds of attacks captured from various IoT devices. By incorporating this dataset into a multimodal training framework, it brings real distribution diversity, which is a test for whether the model has learnt anything generalizable.

2.8 Research Gaps This Thesis Addresses

Reading the literature carefully reveals several gaps that directly motivated the work in this thesis.

1. Studies of CICIDS2017 have focused on testing either of these two machine learning approaches, rather than comparing the two together under similar conditions. A comparative study between RF, XGBoost, and MLP with equal preprocessing has never been done before.
2. Robustness assessments of NIDS against adversarial attacks nearly exclusively consider only one kind of model. There has been no prior work conducted that systematically compares white box gradient attacks to black box transfer attacks for neural nets and tree ensembles, respectively, on CICIDS2017.

3. The FNR – which is really what’s important in terms of missed attacks is almost never mentioned in studies where attack magnitude was varied over a wide range. The saturation phenomenon that occurs with trees during transfer attacks is entirely new.
4. There is currently no unified pipeline to incorporate CICIDS2017, CSE-CICIDS2018, and TON IoT datasets to train models while using identical conditions. [20, 21].
5. .Even though it is recommended to use small flow-based feature sets [21, 24], there is no empirical evidence suggesting that a small set of only eight features is enough for consistent detections across three diverse datasets.

CHAPTER 3

PROBLEM STATEMENT AND RESEARCH OBJECTIVES

3.1 Motivation

This body of work paints a seemingly positive picture that, on closer inspection, reveals a darker reality. The standard for ML-based NIDS trained using CICIDS2017 is to yield AUC scores greater than 0.99, a value cited and used in comparisons to show that the detection task is well done. However, these results depend upon assumptions that are not always met in real-world applications.

The first such assumption is that the problem of class imbalance has been appropriately addressed. If benign flows occur more frequently than attacks, a classifier that is biased towards the majority class will work well according to any statistic but fail miserably at its task of attack detection. In order to achieve this requirement, particular attention should be paid to preprocessing, a topic not very common in literature.

The second assumption made by the authors is that the test set should represent statistically the training set, which implies that an attack on the deployed classifier would mean that attackers possess some knowledge about the classifier itself. In this case, the statistical assumption no longer holds an attack flow may be modified in a small yet deliberate way to make it appear as benign. An NIDS with 99% accuracy on clean data could fail in detecting half the attacks if subject to adversarial manipulation.

Thirdly, there is an assumption that once a model is trained using one type of network environment, the model generalises to other environments. However, traffic characteristics, attack patterns, and feature correlations tend to differ significantly from one network environment to another. A model trained using enterprise traffic in one organisation in 2017 cannot therefore be used on IoT traffic in a different environment without verification. These assumptions lead to specific research problems. This thesis tackles all three.

3.2 Problem Formulation

3.2.1 Stage 1: Building a Trustworthy Baseline

Assume we have a labelled dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ containing N samples, where $\mathbf{x}_i$ is a d -dimensional feature vector and y_i is its binary label, $y_i \in \{0, 1\}$ (0 for a benign sample, 1 for an attack sample), such that there is a class imbalance, i.e., $|\{i : y_i = 0\}| \gg |\{i : y_i = 1\}|$.

Baseline Problem Statement We need to develop a classifier $f_\theta : \mathbb{R}^d \rightarrow \{0, 1\}$ with a good recall rate in respect to attacks while keeping the false positive rate at a satisfactory level:

$$\theta^* = \arg \max_{\theta} \text{F1}(f_\theta, \mathcal{D}_{\text{test}}) \quad \text{subject to} \quad \text{FPR}(f_\theta) \leq \delta \quad (3.1)$$

Apart from training a well-performing classifier, our task also includes establishing a reproducible pipeline for a comparative evaluation of RF, XGBoost, and MLP model families. This means identical preprocessing for all three, careful SMOTE application only on training data, and honest reporting of per-class metrics.

3.2.2 Stage 2: Probing Adversarial Vulnerability

Given a trained classifier f_θ with high clean performance, the adversarial evasion problem asks whether a bounded perturbation to an attack-class input can cause misclassification:

$$f_\theta(\mathbf{x} + \boldsymbol{\delta}) = 0 \quad \text{subject to} \quad \|\boldsymbol{\delta}\|_\infty \leq \varepsilon \quad (3.2)$$

Here $\mathbf{x}$ is a clean attack flow with true label $y = 1$, and the adversary's goal is to push the prediction to 0 (benign) without exceeding the perturbation budget ε .

Two threat models are considered. In the white-box setting, the adversary has full access to the MLP's weights and gradients and can compute the optimal perturbation direction directly. In the black-box transfer setting, the adversary has no access to RF or XGBoost internally, but can generate perturbations against the MLP as a surrogate and apply them to the tree-based targets.

The defence problem is the reverse: given that f_θ is vulnerable to Equation (3.2), find modified parameters θ' such that robustness improves substantially while clean accuracy is preserved:

$$f_{\theta'}(\mathbf{x} + \boldsymbol{\delta}) = 1 \quad \forall \|\boldsymbol{\delta}\|_\infty \leq \varepsilon, \quad \text{Acc}(f_{\theta'}) \approx \text{Acc}(f_\theta) \quad (3.3)$$

3.2.3 Stage 3: Cross-Domain Generalisation

Let $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ be three datasets collected in different network environments with potentially different native feature spaces $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$.

The feature alignment problem is to identify a shared subspace $\mathcal{F}^* \subseteq \mathcal{F}_1 \cap \mathcal{F}_2 \cap \mathcal{F}_3$ such that the projected datasets preserve enough discriminative information for binary classification while remaining compact enough for efficient training.

The cross-domain detection problem is then:

$$\phi^* = \arg \max_{\phi} \text{F1}(g_\phi, \mathcal{D}_{\text{test}}^*) \quad (3.4)$$

where $\mathcal{D}^* = \mathcal{D}_1^* \cup \mathcal{D}_2^* \cup \mathcal{D}_3^*$ is the merged corpus projected onto $\mathcal{F}^*$, and g_ϕ is trained on this unified representation. The question is whether four deep learning architectures — CNN, LSTM, GRU, Transformer — can achieve reliable detection across all three source environments using only $|\mathcal{F}^*| = 8$ shared features.

3.3 Research Questions

These problem formulations give rise to the following research questions:

RQ1. Performance of the baseline models on CICIDS2017 dataset: What is the performance of RF, XGBoost, and MLP in terms of detection accuracy, FNR, F1, and AUC under strict and reproducible preprocessing conditions? What flow level features make the most contribution to classification, and what does that tell us about the detection problem?

RQ2. Vulnerability to adversarial attacks of the baseline models: How does the FNR of each family of models change when perturbation level ε varies from 0.01 to 0.30 for both FGSM and PGD attacks? Are there differences between the

vulnerability patterns of gradient-based and tree-based classifiers under transfer attack?

- RQ3. Effectiveness of defence techniques:** Is adversarial retraining effective for defending MLP? But how much do we pay for it? How much higher is the FNR of our defended classification compared to our baseline?
- RQ4. Attack detection using compact features in cross-domain data:** Is it possible for a deep learning system to develop an accurate binary classification of attacks on the basis of only eight features per flow extracted from the datasets CICIDS2017, CSE-CICIDS2018, and TON IoT?

CHAPTER 4

METHODOLOGY

4.1 Overview

In this chapter we discuss the methodological choices made at each of the three stages of research. The layout of the chapter mirrors the research stages as follows first we describe the preprocessing stage, common to both Stages 1 and 2 then we present the model development process for both the ensembles and deep neural networks then we introduce the attack/defence paradigm used in Stage 2 and finally we discuss the criteria used in evaluating the systems throughout the research stages. Where the methodology differs between stages, the differences are explained explicitly.

4.2 Data Preprocessing

4.2.1 CICIDS2017 Pipeline (Stages 1 and 2)

The CICIDS2017 dataset comes unprocessed in the form of seven CSV files per day output by CICFlowMeter. The initial task is to load and concatenate the seven files. Leading white space in feature names added by CICFlowMeter gets removed at this stage.

Next, there are several cleaning operations to perform. Infinite values, which happen when calculating rates over zero duration flows, get replaced by NaNs and then filled by the respective feature means. Importantly, the statistics for feature means should be calculated only on the training subset and used for filling missing values in both train and test sets. Calculating the means over the entire dataset would amount to data leakage and cause test scores to slightly overestimate the model performance. Duplicates get removed. Non-numeric ID columns like IP addresses are filtered out. Multi-class labels are merged into two categories all non-BENIGN classes are treated as attacks (label 1).

Features are normalized using robust scaling after clipping extreme outliers at the 1st and 99th percentiles. Robust scaling uses the median and interquartile range (IQR)

$$x'_j = \frac{x_j - \text{median}(x_j)}{\text{IQR}(x_j)} \quad (4.1)$$

The scaling statistics are computed using the training set alone. After splitting, class-weighted optimization is applied during training to mitigate imbalance without altering the original data distribution.

$$\mathbf{x}_{\text{new}} = \mathbf{x}_i + \lambda \cdot (\mathbf{x}_{i,\text{nn}} - \mathbf{x}_i), \quad \lambda \sim \mathcal{U}(0, 1) \quad (4.2)$$

where $\mathbf{x}_{i,\text{nn}}$ is a randomly selected nearest neighbour of $\mathbf{x}_i$ among minority-class training samples. The test set remains entirely untouched to retain the natural class distribution.

The train-test ratio is 80-20, and the stratification is done on the basis of class labels, using a fixed random seed for reproducibility.

4.2.2 Multi-Dataset Pipeline (Stage 3)

The third stage consists of three datasets CICIDS2017, CSE-CICIDS2018, and TON IoT. Cleaning of each dataset is done individually through the process mentioned above. After cleaning, the first merged corpus had about five million entries.

The key challenge is feature compatibility. The three datasets were collected in different environments using different tools, and their native feature spaces overlap only partially. The approach taken here was to identify eight flow-level features that are semantically consistent across all three datasets Flow Bytes/s, Flow Packets/s, Protocol, Total Forward Packets, Total Backward Packets, Total Length of Forward Packets, Total Length of Backward Packets. These describe the volume and timing characteristics of network flows in terms that generalise across environments.

For this larger and more heterogeneous dataset, standard scaling is replaced with robust scaling:

$$x'_j = \frac{x_j - \text{median}(x_j)}{\text{IQR}(x_j)} \quad (4.3)$$

Robust scaling is less sensitive to extreme outlier values, which become more common when merging datasets from different network environments with different traffic characteristics.

At the scale of two million records, SMOTE is computationally impractical. Instead, class-weighted binary cross-entropy is used during training. The weights are:

$$w_c = \frac{N}{2 \cdot N_c} \quad (4.4)$$

giving benign weight 0.6563 and attack weight 2.0992. This ensures that minority attack samples receive proportionally more weight in the loss function without requiring any oversampling.

4.3 Stage 1: Ensemble and Neural Network Classifiers

Three model families are trained and evaluated. The objective is to conduct this comparison under truly identical circumstances in order to ensure that the differences arise from the models themselves rather than in their setups.

Random Forest uses 200 trees with balanced class weights and Gini impurity as the split criterion. The balanced class weight setting adjusts sample weights inversely proportional to class frequency, providing an additional layer of imbalance mitigation beyond SMOTE.

XGBoost uses 200 estimators with the histogram-based split-finding method and CUDA acceleration. Early stopping is applied on a validation split to prevent overfitting.

The MLP has two hidden layers — Dense(256, ReLU) followed by Dropout(0.3), then Dense(128, ReLU) followed by Dropout(0.2), with a sigmoid output unit. It is trained with Adam, binary cross-entropy loss, batch size 512, and early stopping on validation loss with patience 5.

Hyperparameters for all three models were tuned on validation splits before finalising, covering tree depth, learning rate, and regularisation for the tree-based models, and layer sizes, dropout rates, and learning rate for the MLP.

Feature importance is extracted from RF using mean decrease in Gini impurity and from XGBoost using gain-based importance. These are reported to identify which flow attributes drive detection and to inform the feature selection decision in Stage 3.

4.4 Stage 3: Deep Learning Architectures

All four architectures receive the same eight-feature input vector and are trained under the same configuration: Adam optimiser with learning rate 10^{-3} , class-weighted BCEWithLogitsLoss, batch size 512, and 10 epochs. PyTorch is used for all Stage 3 experiments.

The CNN applies one-dimensional convolutions over the feature vector, treating it as a spatial sequence. It uses a Conv1D layer (64 filters, kernel size 3), MaxPooling, Flatten, and a Dense(64, ReLU) layer before the sigmoid output. The motivation is that local correlations between adjacent flow features may carry useful structural information.

The LSTM treats the input as a temporal sequence, processing each feature dimension as a timestep.

$$\mathbf{f}_t = \sigma(\mathbf{W}_f[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_f) \quad (4.5)$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_i) \quad (4.6)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c[\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_c) \quad (4.7)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \quad (4.8)$$

By integrating the functions of the gates, the GRU makes the process much simpler:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z[\mathbf{h}_{t-1}, \mathbf{x}_t]) \quad (4.9)$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r[\mathbf{h}_{t-1}, \mathbf{x}_t]) \quad (4.10)$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h[\mathbf{r}_t \odot \mathbf{h}_{t-1}, \mathbf{x}_t]) \quad (4.11)$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \quad (4.12)$$

With fewer parameters involved, the GRU is quicker to learn, and less likely to suffer from overfitting when dealing with the unified corpus.

The Transformer uses multi-head self-attention to model relationships across all feature positions simultaneously:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \quad (4.13)$$

A single encoder block with two attention heads is followed by a feedforward sublayer and a sigmoid output. Class-weighted loss is particularly important for stable transformer convergence on imbalanced tabular data.

4.5 Stage 2: Adversarial Attacks and Defence

4.5.1 Threat Model

Two attacker capability levels are modelled. The white-box attacker has complete access to the MLP’s architecture, weights, and preprocessing transforms, and uses TensorFlow’s automatic differentiation to compute gradient-guided perturbations. This is the strongest realistic threat scenario — it applies in cases where the IDS specification has been leaked or reverse-engineered.

The black-box transfer attacker has no access to the RF or XGBoost classifiers but trains a local MLP surrogate and generates FGSM perturbations against it. These are then applied to the target models without modification. The attack exploits the transferability of gradient-guided perturbations across model families [28].

In both cases, the adversary wants attack-class flows ($y = 1$) to be predicted as benign ($\hat{y} = 0$), with perturbations bounded by:

$$\|\delta\|_\infty \leq \varepsilon \quad (4.14)$$

Five values of ε are tested: $\{0.01, 0.30, 0.10, 0.20, 0.30\}$.

4.5.2 Attack Algorithms

FGSM [6] computes adversarial examples in a single gradient step:

$$\mathbf{x}_{\text{adv}} = \mathbf{x} + \varepsilon \cdot \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}, y)) \quad (4.15)$$

PGD [27] iterates this process over K steps with step size α , projecting back onto the ε -ball after each step:

$$\mathbf{x}^{(k+1)} = \Pi_{\mathbf{x}+\mathcal{S}}\left(\mathbf{x}^{(k)} + \alpha \cdot \text{sign}\left(\nabla_{\mathbf{x}} \mathcal{L}(\theta, \mathbf{x}^{(k)}, y)\right)\right) \quad (4.16)$$

Throughout this work $K = 10$ and $\alpha = 0.01$.

4.5.3 Adversarial Training Defence

Adversarial training [27] mixes clean and adversarially perturbed examples during optimisation:

$$\min_{\theta} \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}} [(1 - \rho) \mathcal{L}(\mathbf{x}, y; \theta) + \rho \mathcal{L}(\mathbf{x}_{\text{adv}}, y; \theta)] \quad (4.17)$$

The mixing ratio $\rho = 0.5$ gives equal weight to clean and adversarial objectives. Training runs for 10 epochs at $\varepsilon = 0.1$ on the SMOTE-balanced training partition.

4.6 Evaluation Metrics

The same metric suite is used across all three stages:

- **Accuracy:** $\text{Acc} = (TP + TN) / (TP + TN + FP + FN)$
- **False Negative Rate:** $\text{FNR} = FN / (FN + TP)$ — the fraction of real attacks that go undetected. This is one of the base measures used in testing IDS because you care much more about the cost of misses versus false positives

- **Precision:** $TP/(TP + FP)$
- **Recall:** $TP/(TP + FN) = 1 - \text{FNR}$
- **F1-Score:** $2 \cdot \text{Prec} \cdot \text{Rec}/(\text{Prec} + \text{Rec})$
- **AUC-ROC:** The area under the receiver operating characteristic curve is the value useful for measuring discrimination without the need for use of an artificial value.

In Stage 2, FNR is recorded at each of the five possible ε values to illustrate the way that the FNR changes depending on the degree of perturbation.

CHAPTER 5

PROPOSED SYSTEM AND MODEL DESIGN

5.1 System Architecture Overview

This thesis is a system comprised not of just one, but three different pipelines, where it is useful to share some infrastructure while others need to diverge due to the differences in the specific research needs for each stage. This was a deliberate design decision, because forcing the three different stages into one pipeline would have meant compromising on something else. Instead, a modular architecture was built where components can be swapped or extended independently.

Table 5.1 maps each system component to the stage it serves. Figure 5.1 shows the five-zone adversarial evaluation pipeline covering Stages 1 and 2. Figure 5.2 shows the multi-dataset integration pipeline for Stage 3.

Table 5.1 System Components and Their Role Across Research Stages

Zone	Component	Function	S1	S2	S3
1a	Single-Dataset Pipeline	CICIDS2017 ingestion, cleaning, scaling, SMOTE	✓	✓	
1b	Multi-Dataset Pipeline	Dataset integration, feature alignment, robust scaling			✓
2a	Ensemble Model Bank	RF, XGBoost, MLP training and serialisation	✓	✓	
2b	Deep Learning Model Bank	CNN, LSTM, GRU, Transformer training and evaluation			✓
3	Attack Module	FGSM and PGD generation, transfer attack		✓	
4	Evaluation Engine	Metric computation, ϵ -sweep, FNR heatmap		✓	
5	Defence Module	Adversarial training, defended model evaluation		✓	

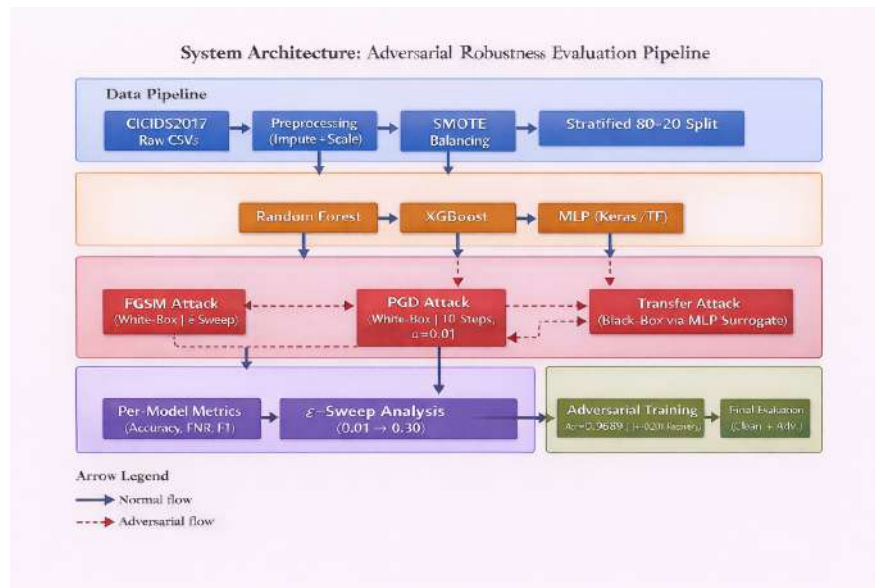


Figure 5.1 End to end adversarial robustness evaluation pipeline

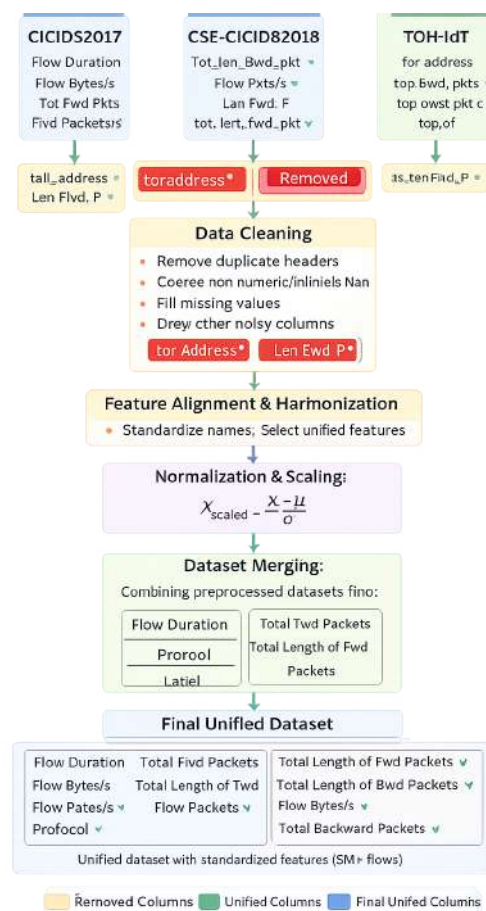


Figure 5.2 Unified preprocessing pipeline for cross domain dataset integration

5.2 Zone 1a: Single-Dataset Pipeline (Stages 1 and 2)

5.2.1 Data Ingestion

Seven daily CSV files from CICIDS2017 are loaded and concatenated. Column names are stripped of whitespace. The concatenated dataset has approximately 2.8 million rows and 79 feature columns plus one label column.

5.2.2 Cleaning and Preparation

Infinite values are replaced with column means. NaN entries are imputed with column means. Both operations use statistics computed on the training partition only. Non-numeric identifier columns are dropped. Labels are mapped to binary: BENIGN becomes 0, everything else becomes 1.

5.2.3 Scaling and Balancing

Extreme outliers are clipped at the 1st and 99th percentiles before robust scaling using median and interquartile range statistics computed on the training partition. Class-weighted optimization is used during training to handle imbalance while preserving the original data distribution.

The 80–20 stratified split with fixed seed (`seed = 42`) is performed before any SMOTE application.

5.3 Zone 1b: Multi-Dataset Pipeline (Stage 3)

5.3.1 Dataset Integration

Three datasets are used:

- **CICIDS2017** — enterprise network traffic, seven capture days, approximately 2.8 million flows, 79 CICFlowMeter features.
- **CSE-CICIDS2018** — extended enterprise traffic, ten capture days, larger scale and additional attack categories, same CICFlowMeter feature space.
- **TON IoT** — IoT and Industrial IoT traffic with substantially different traffic characteristics and attack modalities from enterprise datasets.

Each dataset is cleaned independently before merging. The initial merged corpus contained approximately 5,041,786 samples. After filtering and subsampling to manageable scale, the final corpus contains 2,000,000 samples.

5.3.2 Feature Alignment

The eight shared features selected for the unified representation are shown in Table 5.2. These describe volume and timing properties of flows that have consistent semantics across all three datasets.

Table 5.2 Shared Feature Set for Cross-Domain Framework

Index	Feature	Description
1	Flow Duration	Total duration of the flow
2	Total Forward Packets	Packet count, forward direction
3	Total Backward Packets	Packet count, backward direction
4	Total Fwd Packet Length	Byte count, forward direction
5	Total Bwd Packet Length	Byte count, backward direction
6	Flow Bytes/s	Bytes per second
7	Flow Packets/s	Packets per second
8	Protocol	Transport protocol identifier

5.3.3 Scaling and Class Weights

RobustScaler is applied using median and IQR computed on training data. Class weights are computed as $w_c = N/(2N_c)$, giving benign weight 0.6563 and attack weight 2.0992.

5.4 Zone 2a: Ensemble and MLP Model Bank (Stages 1 and 2)

5.4.1 Random Forest Configuration

Table 5.3 Random Forest Hyperparameters

Parameter	Value
Number of estimators	200
Class weight	balanced
Criterion	Gini impurity
Maximum features	$\sqrt{d}$
Random state	42

RF is non-differentiable, so gradient-based attacks cannot be applied to it directly. It is instead evaluated against transfer attacks generated using the MLP as a surrogate.

5.4.2 XGBoost Configuration

Table 5.4 XGBoost Hyperparameters

Parameter	Value
Number of estimators	200
Tree method	hist
Device	CUDA
Learning rate	0.1
Max depth	6
Subsample	0.8
Column sample per tree	0.8

5.4.3 MLP Architecture

Table 5.5 MLP Architecture

Layer	Type	Units/Rate	Activation
1	Dense	256	ReLU
2	Dropout	0.3	—
3	Dense	128	ReLU
4	Dropout	0.2	—
5	Dense (output)	1	Sigmoid

The MLP is implemented in Keras/TensorFlow. Its differentiability makes it the target for white-box FGSM and PGD attacks in Stage 2, and its trained weights serve as the surrogate for generating transfer attacks against RF and XGBoost. Models are serialised after training and reloaded for adversarial evaluation, with a verification step confirming that reload introduces no measurable accuracy degradation.

5.4.4 Feature Importance

Feature importance is extracted using mean decrease in Gini impurity for RF and gain-based importance for XGBoost. The top-10 features are reported and their connection to adversarial perturbability is discussed in Chapter 7.

5.5 Zone 2b: Deep Learning Model Bank (Stage 3)

All four architectures share the training configuration in Table 5.6.

Table 5.6 Shared Deep Learning Training Configuration

Parameter	Value
Framework	PyTorch
Optimiser	Adam, lr = 10^{-3}
Loss	Class-weighted BCEWithLogitsLoss
Batch size	512
Epochs	10

The CNN applies Conv1D (64 filters, kernel 3) followed by MaxPooling, Flatten, Dense(64, ReLU), and sigmoid output.

The LSTM and GRU treat the eight features as a sequence, with the final hidden state passed through a sigmoid output layer. The GRU uses fewer parameters than the LSTM and is generally faster to train on this dataset.

The Transformer uses a single encoder block with two attention heads and a position-wise feedforward sublayer before the sigmoid output. All four architectures are initialised with the same random seed to ensure that performance differences reflect architectural choices rather than initialisation variance.

5.6 Zones 3–5: Adversarial Evaluation and Defence (Stage 2)

5.6.1 Attack Generation (Zone 3)

Adversarial examples are generated in batches of 2,048 using TensorFlow’s GradientTape API. For FGSM, a single forward and backward pass is computed per batch. For PGD, the loop runs for $K = 10$ iterations with step size $\alpha = 0.01$, with projection back onto the ε -ball after each step.

Transfer attacks apply the FGSM examples from the MLP directly to RF and XGBoost without modification.

5.6.2 Evaluation Engine (Zone 4)

The evaluation engine computes accuracy, FNR, precision, recall, and F1-score for each model under each attack condition at each ε value. Results are visualised as clustered bar charts, FNR-vs- ε line plots, and FNR heatmaps. All evaluation uses the same test partition as Stage 1.

5.6.3 Defence Module (Zone 5)

The adversarial training defence trains a new MLP instance from scratch on the SMOTE-balanced training partition, with each mini-batch split equally between clean and FGSM-perturbed examples at $\varepsilon = 0.1$. The defended model is then evaluated on both the clean test set and the FGSM adversarial test set.

CHAPTER 6

EXPERIMENTAL SETUP

6.1 Overview

This chapter describes the datasets, compute environment, training configurations, and evaluation protocols used across all three research stages. Where setup is shared between stages it is described once. Stage-specific details are given in separate subsections.

6.2 Datasets

6.2.1 CICIDS2017 (Stages 1 and 2)

CICIDS2017 [34] was generated at the "Canadian Institute for Cybersecurity, University of New Brunswick". It covers seven days of traffic in a controlled lab network designed to simulate a realistic enterprise environment, with genuine attack execution on each capture day. The dataset contains approximately 2.83 million flow records described by 79 CICFlowMeter features.

Table 6.1 CICIDS2017 Dataset Statistics

Property	Value
Total raw flow records	≈ 2.83 million
Features per flow	79 (CICFlowMeter)
Capture days	7
Attack categories	DDoS, PortScan, Web Attack, Infiltration, Botnet, Heartbleed
Working subset (20% sample)	566,149 rows
Training rows after SMOTE	681,642
Test rows (stratified hold-out)	141,538
Benign:Attack ratio in test	$\approx 4:1$

Given that the amount of available memory for loading the data was limited (12.7 GB system memory), it was decided to sample the data by 20% utilizing stratified random sampling. This kind of sampling is justified considering previous experience with this specific dataset [3].

Table 6.2 Attack Category Distribution in CICIDS2017 (Sampled Subset)

Category	Count	% of Attacks
DDoS	128,027	34.7
PortScan	158,930	43.1
Web Attack – Brute Force	7,938	2.2
Web Attack – XSS	2,180	0.6
Web Attack – SQL Injection	21	0.01
Infiltration	36	0.01
Botnet	1,966	0.5
Heartbleed	11	0.003
DoS (various)	68,618	18.6
Total Attacks	368,727	100.0

With DDoS and PortScan making up over 77%, the model performance evaluation will be highly skewed toward these two categories. Web and Infiltration attacks are found in very few samples, making them more difficult to catch and less likely to be detected — this should be kept in mind when evaluating recall.

6.2.2 CSE-CICIDS2018 (Stage 3)

The CSE-CICIDS2018 follows the exact method of collection used for the previous year’s dataset but for ten days’ worth of data. The CICFlowMeter pipeline used remains unchanged, thus ensuring full compatibility for feature alignment. It includes new kinds of attacks that were absent in the previous version, including large scale Infiltrations and Bots.

6.2.3 TON IoT (Stage 3)

TON IoT involves the capture of Internet of Things network traffic for multiple platforms. It contains attacks involving seven different kinds of threats, all of which differ greatly from the traffic characteristics in corporate networks. Including it in the unified training corpus is what makes Stage 3 a genuine cross-domain exercise rather than just a multi-enterprise experiment.

6.2.4 Unified Corpus (Stage 3)

Table 6.3 Unified Multi-Dataset Corpus Statistics

Property	Value
Initial merged size	$\approx 5,041,786$ samples
Final corpus after cleaning	2,000,000 samples
Shared features	8 (flow-level)
Labels	Binary (0 = benign, 1 = attack)
Benign class weight	0.6563
Attack class weight	2.0992
Train–test split	80–20 stratified

6.3 Compute Environment

Table 6.4 Compute Environment

Component	Specification
Platform	Google Colaboratory
GPU	NVIDIA Tesla T4, 16 GB VRAM
System RAM	12.7 GB
Deep learning (S1–S2)	TensorFlow 2.x with CUDA 11
Deep learning (S3)	PyTorch with CUDA
ML library	Scikit-learn 1.x
Gradient boosting	XGBoost 1.x (CUDA)
Oversampling	imbalanced-learn (SMOTE)
Python	3.10

Adversarial example generation in Stage 2 uses TensorFlow’s GradientTape for GPU-accelerated gradient computation, with batch size 2,048 to avoid GPU memory overflow. XGBoost training uses the hist tree method with CUDA acceleration. RF and XGBoost inference runs on CPU.

6.4 Model Training Details

6.4.1 Stage 1: Ensemble Classifiers and MLP

RF was trained with scikit-learn’s RandomForestClassifier with 200 trees and balanced class weights. Hyperparameters were selected based on prior literature and validated on a 15% internal validation split.

XGBoost was trained with the XGBClassifier interface using GPU acceleration and early stopping (patience 10) on the same 15% validation split from the SMOTE-balanced training data.

The MLP was trained in Keras with early stopping (patience 5 on validation loss). The best checkpoint was saved and reloaded for Stage 2 adversarial evaluation. Before

any adversarial experiment, the reloaded models are verified to reproduce training-time AUC values within ± 0.001 .

6.4.2 Stage 2: Adversarial Training Defence

No retraining of Stage 1 base models occurs during adversarial evaluation. Only the adversarial training defence involves training from scratch — a new MLP instance is trained on the SMOTE-balanced training partition with the mixed clean/adversarial objective.

Table 6.5 Adversarial Training Configuration

Parameter	Value
Attack used	FGSM at $\varepsilon = 0.1$
Mixing ratio ρ	0.5
Training epochs	10
Batch size	2,048 (GPU limit)
Total pipeline runtime	≈ 7.3 minutes

6.4.3 Stage 3: Deep Learning on Unified Corpus

All four architectures (CNN, LSTM, GRU, Transformer) use the configuration in Table 6.6. The same random seed and the same training partition are used for all four, so performance differences are attributable to architecture alone.

Table 6.6 Deep Learning Training Configuration (Stage 3)

Parameter	Value
Optimiser	Adam, $\text{lr} = 10^{-3}$
Loss	Class-weighted BCEWithLogitsLoss
Batch size	512
Epochs	10
Framework	PyTorch

6.5 Evaluation Protocols

6.5.1 Stage 1

All metrics are computed on the stratified hold-out test partition (141,538 samples). Confusion matrices and ROC curves are generated for all three models. Feature importance is extracted and the top-10 features are reported.

6.5.2 Stage 2

Stage 1 model checkpoints are loaded and verified before any adversarial experiment. For each $\varepsilon \in \{0.01, 0.05, 0.10, 0.20, 0.30\}$:

1. FGSM and PGD adversarial test sets are generated for the MLP.

2. FGSM examples are applied as transfer attacks to RF and XGBoost.
3. Accuracy, FNR, precision, recall, and F1 are computed for all models under all attack conditions.

FNR-vs- ϵ plots and FNR heatmaps comparing FGSM and PGD are generated. The adversarial training defence is then evaluated on both clean and adversarial test partitions.

6.5.3 Stage 3

All four architectures are trained and evaluated on the same unified corpus partition. Accuracy, F1, precision, recall, and AUC are reported for each. Training convergence curves, ROC curves, precision-recall curves, and confusion matrices are generated.

6.6 Reproducibility

A fixed random seed (`seed = 42`) is used throughout for dataset splitting, SMOTE sampling, model initialisation, and adversarial example generation. Stage 1 models are saved after training and reloaded from checkpoints for Stage 2 experiments. All experiments are implemented in Jupyter notebooks with cell-level documentation, enabling end-to-end reproduction.

CHAPTER 7

RESULTS AND DISCUSSION

7.1 Overview

This chapter presents results from all three research stages and discusses what they mean. Section 7.2 covers the Stage 1 baseline evaluation on CICIDS2017. Section 7.3 covers the Stage 2 adversarial robustness experiments. Section 7.4 covers the Stage 3 cross-domain deep learning results. Section 7.5 integrates the results of this study and responds to the research questions posed in Chapter 3.

7.2 Stage 1: Baseline Performance on CICIDS2017

7.2.1 Quantitative Results

Table 7.1 below summarizes the performance metrics of all three algorithms based on the test partition. Note that all three algorithms were trained using the same SMOTE-balanced dataset processed via the same preprocessing pipeline.

Table 7.1 Baseline Model Performance on CICIDS2017 Test Partition

Model	Accuracy	Precision	Recall	F1	AUC
Random Forest	0.9987	0.9970	0.9981	0.9975	0.9998
XGBoost	0.9991	0.9960	0.9995	0.9977	0.9950
MLP	0.9728	0.9820	0.9929	0.9874	0.9980

XGBoost exhibits superior performance by multiple metrics: AUC = 0.995 and FNR of just 0.05% for clean traffic. RF performs similarly with AUC = 0.9998 and FNR = 0.30%. MLP lags a bit behind with AUC = 0.9980 and FNR = 0.71%. This difference in performance is in line with previous research on IDS, where gradient boosting typically outperforms feedforward neural networks when applied to tabular flow data sets.

The FNR metrics deserve extra attention. Even a 0.71% difference means missing 198 attacks on a test partition containing 141,538 samples. While this discrepancy does not sound too terrible per se, it becomes the starting point from which Stage 2 evaluates model degradation. And the contrast becomes apparent.

7.2.2 ROC Analysis

All three algorithms demonstrate nearly perfect separability across different thresholds: AUC $\geq$ 0.997. Specifically, the ROC curve of XGBoost increases in the top-left corner the most, meaning the algorithm is able to generate more true positives at relatively low FPR levels.

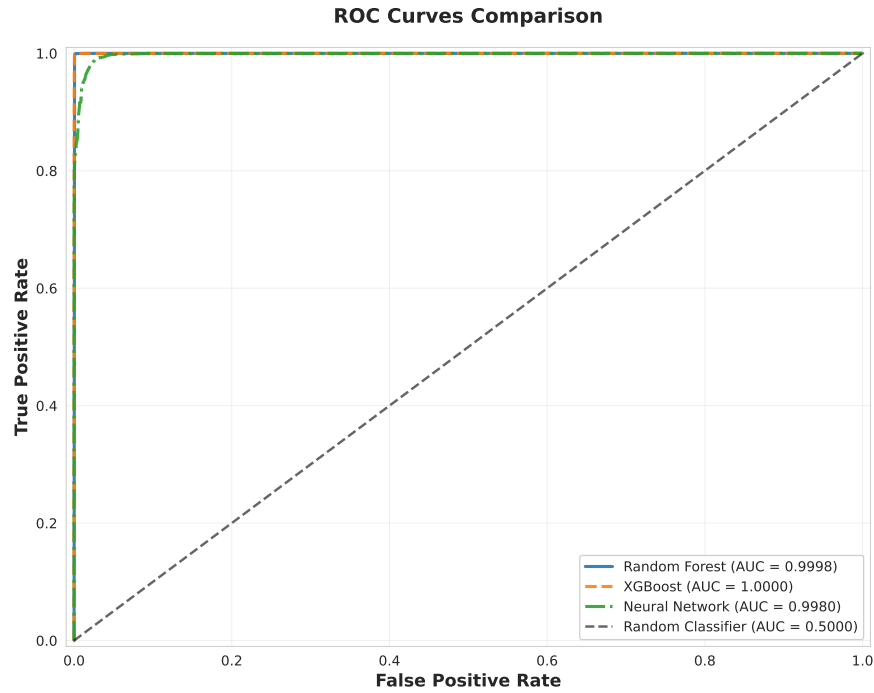


Figure 7.1 ROC curves for RF, XGBoost, and MLP on the CICIDS2017 test partition. AUC values are annotated in the legend.

7.2.3 Confusion Matrix Analysis

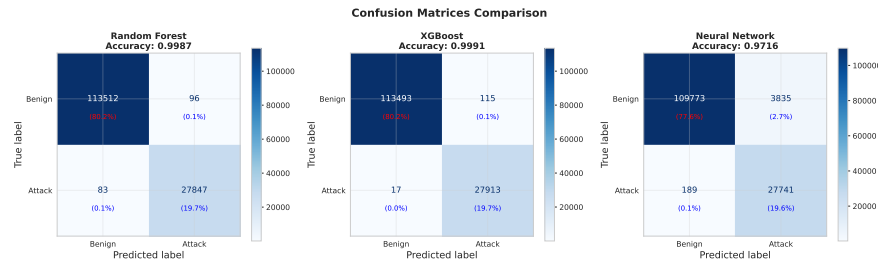


Figure 7.2 Confusion matrices for RF, XGBoost, and MLP on the CICIDS2017 test partition.

XGBoost produces just 17 false negatives on the test set. RF produces 83. The MLP produces 189 — more than ten times XGBoost’s number. This is before any adversarial perturbation. The gap matters because the MLP, as the only differentiable model, is also the one that will be attacked in Stage 2. Starting from a higher FNR baseline means adversarial degradation is measured against a weaker starting point.

7.2.4 Feature Importance

Table 7.2 Top-10 Features by RF Importance

Rank	Feature	Importance
1	Flow Duration	0.112
2	Packet Length Mean	0.098
3	Fwd Packet Length Max	0.087
4	Bwd Packet Length Max	0.076
5	Flow Bytes/s	0.071
6	Fwd IAT Total	0.065
7	Bwd IAT Total	0.059
8	Fwd Packet Length Mean	0.055
9	Init Win Bytes Forward	0.048
10	Packet Length Std	0.043
Top-10 Cumulative		0.714

Flow Duration, Packet Length Mean, and Fwd Packet Length Max are the three most important features — together they account for nearly 30% of RF’s predictive power. The top-10 together account for 71.4%. This suggests that a much smaller feature set than the full 79 would be sufficient for strong detection, which is one of the observations that motivated the eight-feature design in Stage 3.

These features also happen to be the ones most perturbed by FGSM — the pitch of the loss with respect to inputs will be largest for features that contribute most to the classification decision, so adversarial perturbations naturally concentrate on the most discriminative attributes.

7.2.5 Stage 1 Summary

RQ1 is answered: XGBoost achieves the best baseline performance (AUC = 0.995, FNR = 0.05%) under a rigorous preprocessing pipeline. All three models achieve AUC above 0.997. Feature importance analysis shows that a compact top-10 subset accounts for 71.4% of model importance, directly motivating the eight-feature cross-domain design in Stage 3.

7.3 Stage 2: Adversarial Robustness Evaluation

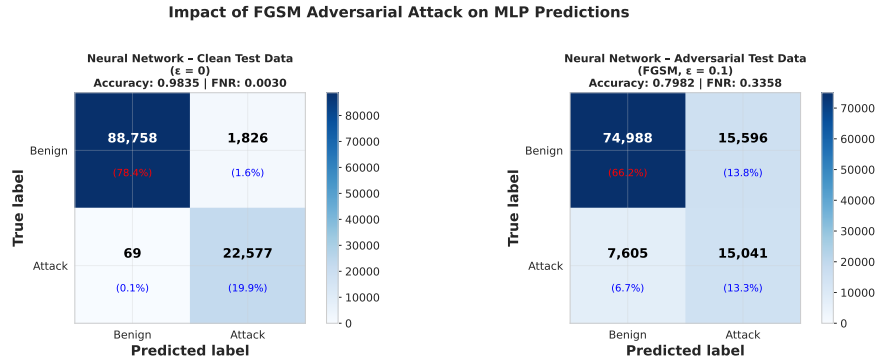
7.3.1 Attack Impact at $\varepsilon = 0.1$

Table 7.3 shows what happens to all three models under FGSM and PGD at the reference perturbation budget $\varepsilon = 0.1$.

Table 7.3 Model Performance Under Adversarial Attack ($\varepsilon = 0.1$)

Model	Condition	Acc.	Prec.	Rec.	FNR
MLP	Clean	0.9835	0.9252	0.9970	0.0030
	FGSM ($\varepsilon = 0.1$)	0.7982	0.4909	0.6642	0.3358
	PGD ($\varepsilon = 0.1$)	0.7822	0.4632	0.6635	0.3365
RF	Clean	0.9987	0.9961	0.9975	0.0025
	FGSM transfer	0.9298	0.9785	0.6581	0.3419
XGBoost	Clean	0.9989	0.9950	0.9996	0.0004
	FGSM transfer	0.9178	0.8965	0.6587	0.3413
MLP (defended)	Clean	0.9635	—	—	—
	FGSM ($\varepsilon = 0.1$)	0.9519	—	—	0.0282

The MLP degradation under FGSM at $\varepsilon = 0.1$ is severe. FNR jumps from 0.30% to 33.58% — a 112-fold increase. False negatives increase from 69 to 7,605 in the confusion matrix. This indicates that there is one out of every three real attacks not detected by this model. Precision will also decrease, meaning that the attack makes benign instances become an attack, but the attacks are missed, which is the operationally important issue.

**Figure 7.3** Confusion matrices for the MLP on unperturbed (left) and FGSM-perturbed (right) test sets for $\varepsilon = 0.1$. The number of false negatives increases from 69 to 7,605.

In contrast, the transfer attacks on RF and XGBoost are also severe. RF's FNR goes from 0.25% to 34.19%, and XGBoost's from 0.04% to 34.13%, despite neither model being directly differentiable. These models were chosen partly because of the intuition that non-differentiable classifiers are safer against gradient-based attacks. The transfer results challenge that intuition directly.

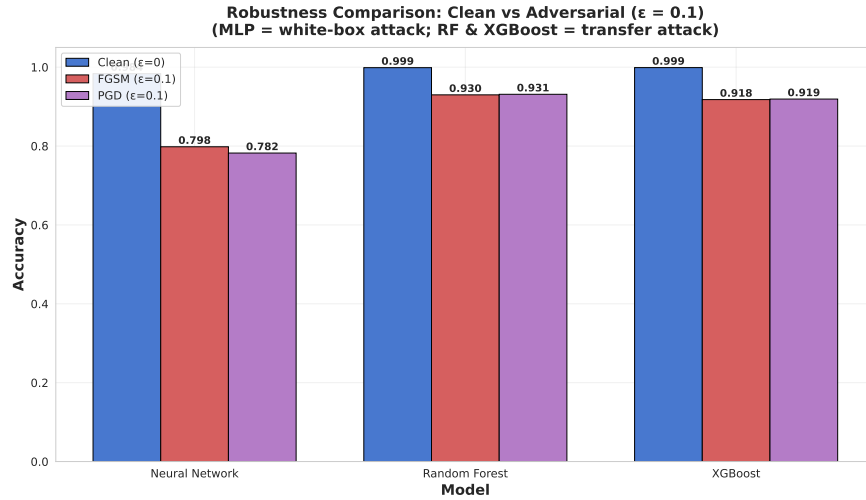


Figure 7.4 Accuracy under clean, FGSM, and PGD conditions for all three models at $\epsilon = 0.1$.

7.3.2 FNR Saturation in Tree-Based Models

The most unexpected finding in Stage 2 is illustrated in Figure 7.5.

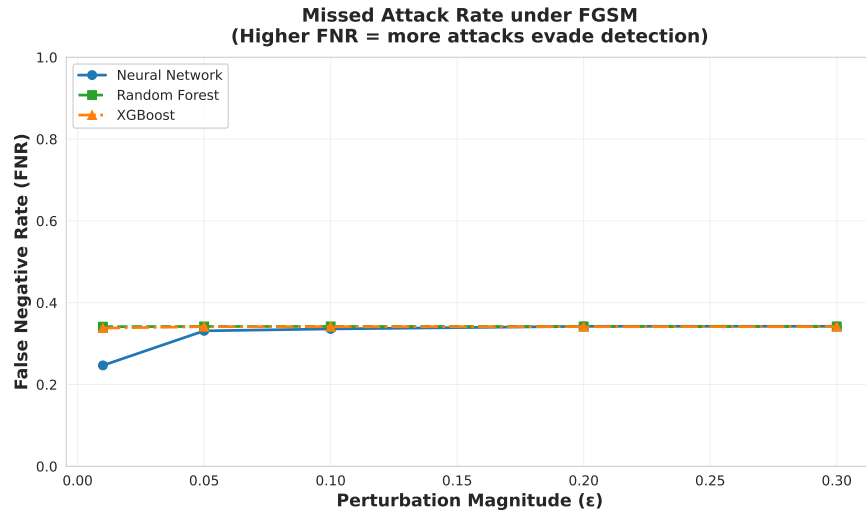


Figure 7.5 FNR versus perturbation magnitude ϵ under FGSM. Tree-based models saturate immediately at $\epsilon = 0.01$; MLP FNR grows monotonically.

RF and XGBoost reach approximately 34% FNR at the smallest tested perturbation budget ($\epsilon = 0.01$) and stay there regardless of how large ϵ gets. The MLP, by contrast, starts at FNR 0.247 at $\epsilon = 0.01$ and climbs to 0.342 at $\epsilon = 0.30$.

Why does this happen? Tree-based classifiers partition the feature space using axis-aligned splits — boundaries that run perpendicular to individual feature axes. A perturbation of size $\epsilon = 0.01$ computed in the FGSM gradient direction for the MLP is already large enough to push some samples across these axis-aligned boundaries. But increasing ϵ further does not help, because the MLP gradient direction does not align

with the additional splits that would need to be crossed to cause more misclassifications. The attack reaches maximum effectiveness at minimal perturbation and cannot improve from there.

The practical implication is sharp: an attacker targeting a tree-based IDS using an MLP surrogate needs only the cheapest possible perturbation to achieve near-maximal evasion. There is no added value to defending from larger perturbations, since the attacks saturate at $\varepsilon = 0.01$.

Table 7.4 FNR Values Across Models, Attack Types, and Perturbation Magnitudes

Attack	Model	ε				
		0.01	0.05	0.10	0.20	0.30
FGSM	MLP	0.247	0.331	0.336	0.342	0.342
	RF	0.341	0.342	0.342	0.342	0.342
	XGBoost	0.338	0.341	0.341	0.341	0.341
PGD	MLP	0.272	0.333	0.337	0.337	0.337
	RF	0.342	0.342	0.342	0.342	0.342
	XGBoost	0.339	0.342	0.342	0.342	0.342

The performance of the iterative PGD attack is better than FGSM only at smaller values of ε , for the MLP network (0.272 vs 0.247 at $\varepsilon = 0.01$). This was expected, as the iterative approach can find a more successful direction for perturbation. For higher values of ε , both attacks will give the same level of FNR. For both RF and XGBoost models, PGD and FGSM are almost identical.

7.3.3 Adversarial Training Results

Table 7.5 Defence Effectiveness: Undefined vs Defended MLP

Condition	Accuracy	FNR
Undefined, clean traffic	0.9835	0.0030
Undefined, FGSM ($\varepsilon = 0.1$)	0.7982	0.3358
Defended, clean traffic	0.9635	—
Defended, FGSM ($\varepsilon = 0.1$)	0.9519	0.0282

Note: The clean accuracy values in this table (MLP: 0.9835, RF: 0.9987, XGBoost: 0.9989) reflect the adversarial evaluation pipeline, which uses a 20% random sample of CICIDS2017 (566,149 observations, fixed seed 42, test partition of 113,230 samples) as described in Section 6.2. The Stage 1 baseline in Table 7.1 uses the full dataset (2.8 million flows, 141,538 test samples). Minor metric differences between the two tables reflect this difference in sample size and random seed, not a change in model weights.

Adversarial training gains back 15.4 percentage points of accuracy in terms of adversarial data, going from 0.7982 to 0.9519. At the same time, adversarial training

suffers a loss of 2.0% in terms of accuracy for clean data, which goes from 0.9835 to 0.9635. The defended FNR in case of adversarial traffic is 2.82%, which shows significant improvement and makes sense to accept in a real-world deployment.

The gap between the defended FNR in case of adversarial traffic (2.82%) and the clean baseline FNR (0.30%) is anticipated, since it corresponds to a known trade-off between the robustness and accuracy of the model in adversarial training [27]. The gap is real but small enough that it does not undermine the case for adversarial training as a practical defence.

7.3.4 Stage 2 Summary

RQ2 and RQ3 are answered. A single FGSM step at $\varepsilon = 0.1$ causes a 112-fold FNR increase in the MLP. Transfer attacks cause immediate FNR saturation at 34% in RF and XGBoost at $\varepsilon = 0.01$ — a novel finding with direct security implications. Adversarial training recovers 15.4 percentage points of adversarial accuracy at under 2% clean accuracy cost.

7.4 Stage 3: Cross-Domain Deep Learning

7.4.1 Training Convergence

All four architectures converge stably over 10 epochs on the unified corpus. Loss during training is shown to monotonically decrease in all cases, and there are no instances of non-convergence or oscillations, an encouraging sign, considering the nature of the data we have.

7.4.2 Performance Comparison

Table 7.6 Deep Learning Architecture Comparison on Unified Corpus

Model	Accuracy (%)	F1 Score
CNN	93.81	0.8646
LSTM	93.83	0.8623
GRU	94.23	0.8714
Transformer	91.50	0.8189

The GRU comes out ahead with 94.23% accuracy and $F1 = 0.8714$. CNN and LSTM are close together at 93.81% and 93.83% — the difference is small enough that it is not meaningful to rank them definitively. The Transformer is the weakest at 91.50%, though it still exceeds 91% and produces stable training behaviour.

All four architectures clearing 91% on a merged corpus from three heterogeneous network environments — enterprise, extended enterprise, and IoT — using only eight shared features is the main result here. It shows that this compact representation captures enough generalised discriminative information to work across very different network contexts.

7.4.3 Cross-Domain Generalization Evaluation

Beyond within-distribution performance, the GRU architecture was evaluated under a leave-one-dataset-out protocol to assess generalization to entirely unseen network environments. In each of three experiments, two datasets were used for training and the third was held out as the test set, with no samples from the test dataset present during training or preprocessing. The RobustScaler was fit exclusively on the training split to prevent data leakage.

Table 7.7 Cross-Domain Evaluation Results Using GRU (Leave-One-Dataset-Out Protocol)

Training Datasets	Test Dataset	Accuracy (%)	Precision	Recall	F1 Score
CICIDS2017 + CSE-CICIDS2018	TON_IoT	54.73	0.5524	0.4993	0.5245
CICIDS2017 + TON_IoT	CSE-CICIDS2018	62.56	0.5899	0.8239	0.6875
CSE-CICIDS2018 + TON_IoT	CICIDS2017	45.72	0.4558	0.4418	0.4487

Cross-domain F1 scores range from 0.45 to 0.69, substantially below the within-distribution F1 of 0.87. This gap confirms that distribution shift across network environments remains a significant challenge even when the feature set is constrained to eight domain-invariant flow statistics. The experiment training on CICIDS2017 + TON_IoT and testing on CSE-CICIDS2018 achieves the highest cross-domain F1 of 0.6875, consistent with the shared enterprise traffic characteristics between those two datasets. The lowest F1 of 0.45 occurs when testing on CICIDS2017 after training on CSE-CICIDS2018 + TON_IoT: although CSE-CICIDS2018 shares the same data generation methodology as CICIDS2017, the addition of TON_IoT traffic in training introduces IoT-specific patterns that shift the learned decision boundary away from the enterprise flow characteristics that distinguish CICIDS2017 samples. These results establish a reproducible cross-domain baseline for this dataset combination and motivate future investigation of domain-adversarial training as a means of improving transfer performance.

7.4.4 Why the GRU Performs Best

The GRU’s lead over the LSTM is modest but consistent. Gated recurrent units model the sequential dependencies between feature dimensions with fewer parameters than LSTMs, which reduces the risk of overfitting on any single source domain. On the kind of compact, cross-domain representation used here, the GRU’s simpler gating mechanism seems to be a better fit than the LSTM’s more complex one.

The Transformer’s lower performance is harder to explain definitively, but a plausible interpretation is that on an eight-dimensional input, the self-attention mechanism simply does not have enough structure to exploit its capacity for long-range dependency modelling. The attention heads may essentially be learning to weight the eight features uniformly, reducing the transformer to something closer to a weighted average with extra computation. This is a limitation that would likely improve with more complex or higher-dimensional input representations.

The CNN’s competitive performance (93.81%) is somewhat surprising given that flow features do not have an obvious spatial structure in the way that image pixels do.

One interpretation is that local correlations between adjacent features in the feature vector carry more information than expected — for instance, forward and backward packet counts are adjacent and correlated, and the CNN may be learning to exploit these local relationships.

7.4.5 Evaluation Curves

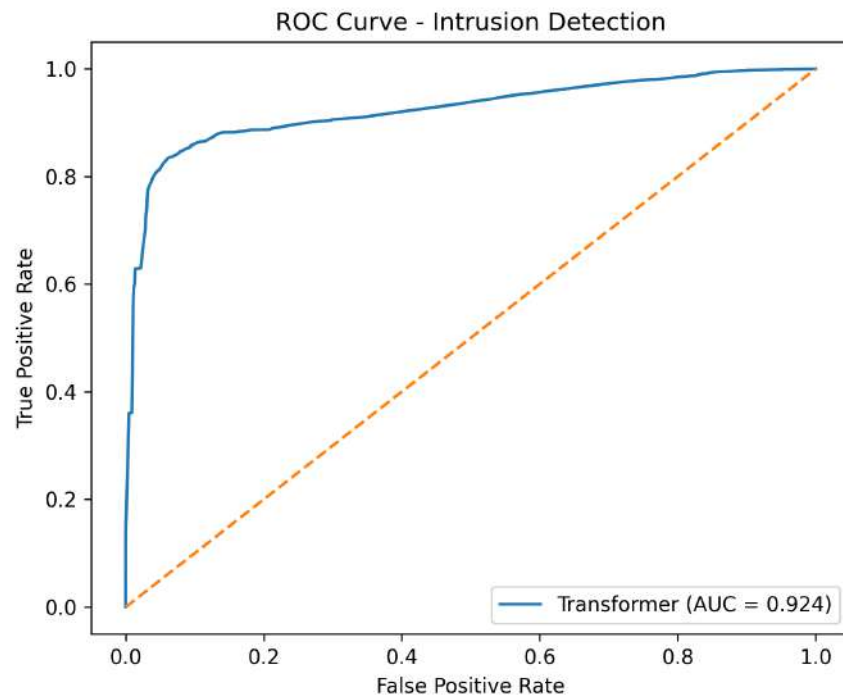


Figure 7.6 ROC curve for the Transformer on the unified test partition (AUC = 0.924).

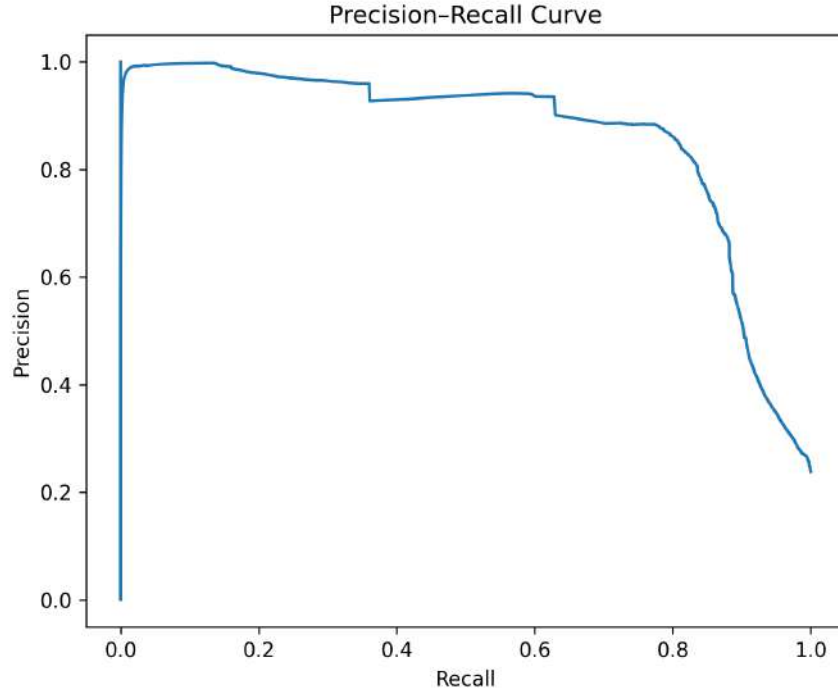


Figure 7.7 Precision-recall curve for the Transformer on the unified test partition.

The ROC and precision-recall curves confirm reliable discrimination across all thresholds. High precision is maintained over a broad range of recall values, indicating that the classifiers are not trading off precision heavily to achieve recall — an important property when false alarms are operationally costly.

7.4.6 Stage 3 Summary

RQ4 is answered. Under within-distribution evaluation, the GRU achieves 94.23% accuracy and $F1 = 0.8714$, with all four architectures exceeding 91%, confirming that compact flow representations are sufficient for reliable cross-domain detection. Under the leave-one-dataset-out protocol, GRU $F1$ scores range from 0.45 to 0.69, establishing a reproducible cross-domain baseline and quantifying the distribution gap between enterprise and IoT network environments.

7.5 Cross-Cutting Discussion

7.5.1 Answers to Research Questions

RQ1: XGBoost achieves the best baseline on CICIDS2017 ($AUC = 0.995$, $FNR = 0.05\%$). The top-10 features account for 71.4% of importance, motivating Stage 3’s compact representation.

RQ2: FGSM and PGD cause a 112-fold FNR increase in the MLP at $\varepsilon = 0.1$. Transfer attacks produce immediate FNR saturation near 34% in RF and XGBoost at $\varepsilon = 0.01$. Tree-based classifiers are not inherently safer than neural networks against adversarial evasion via transfer.

RQ3: Adversarial training restores 15.4 percent points of adversarial accuracy for just

under 2% loss in clean accuracy. Simple and efficient without the need for complicated ensembles.

RQ4: A GRU model trained on eight shared features achieves 94.23% accuracy within-distribution. Leave-one-dataset-out evaluation yields F1 scores of 0.45 to 0.69, establishing that compact flow features support within-distribution detection but that cross-domain generalization to fully unseen environments remains an open challenge.

7.5.2 Connecting the Three Stages

The three phases may be more related than one first suspects. The result of the feature importance analysis from Stage 1, where ten features contribute to 71% of the predictive strength, clearly inspires the eight-feature design of Stage 3. If all 79 features were equally important, a compact shared representation would have been harder to justify.

The adversarial saturation finding in Stage 2 has implications for Stage 3 as well. The eight-feature representation used across all datasets may be less susceptible to transfer attacks than a richer representation tied to one specific dataset’s feature distribution — an interesting direction for future adversarial evaluation of cross-domain models.

7.5.3 Broader Observations

Three things stand out from looking at all three stages together.

Benchmark AUC values above 0.99 are best-case figures that apply only to same-distribution test data. Stage 2 shows how dramatically adversarial perturbation changes the picture. Stage 3 shows that cross-domain deployment changes it too, though in a different direction — the question is not just whether the model can be fooled, but whether the patterns it learned generalise.

The assumption that non-differentiable classifiers are safer against adversarial attacks is not supported by the transfer attack results. XGBoost’s FNR rises from 0.05% to 34.3% at $\varepsilon = 0.01$. Security experts opting for decision tree models owing to their opacity should reconsider their choice.

Eight flow-level features suffice for effective cross-domain detection. This is a practical contribution; it indicates that lightweight, standardised IDS setups are possible without complex dataset-specific feature engineering.

CHAPTER 8

CONCLUSION, FUTURE SCOPE AND SOCIAL IMPACT

8.1 Conclusion

This thesis aimed to solve three underlying issues related to the impressive results presented in benchmarks for network intrusion detection systems based on ML: dependency of the results on the data preprocessing, their uselessness in adversarial settings, and inability to generalize these results beyond the environment used to train a model. The research followed an interconnected process consisting of three stages.

Stage one created a replicable baseline on the dataset CICIDS2017. The pipeline included all steps starting from loading raw CSV files, going through imputation, normalization, SMOTE oversampling, and stratified splitting. It then trained three ML models – Random Forest, XGBoost, and MLP – under identical conditions. XGBoost outperformed others with AUC = 0.995 and FNR = 0.05% in terms of clean test data. Feature selection revealed that ten flow-level features contribute to 71.4% of total predictive value. This result will prove to be particularly useful in stage three.

Stage two of our analysis focused on determining how such high-performing models perform when under attack. Applying a single FGSM step with an attack strength of $\varepsilon = 0.1$ is sufficient for the MLP to increase its false negative rate by 112 times, up from 0.30% to 33.58%. Random Forest and XGBoost transfer attacks present an even more frightening picture in terms of their resistance to FGSM; both classifiers hit around 34% of false negatives for the minimum attacker budget $\varepsilon = 0.01$, after which point the number of missed detections no longer changes despite the increase in ε . This saturation effect stems from the decision boundary axis alignment of trees and their inability to resist FGSM attacks beyond a certain budget due to their linear nature in that direction. It was not previously established that decision tree-based classifiers possess such poor resistance capabilities relative to neural networks, and represents an interesting discovery in this regard. Finally, adversarial training was found to improve the MLP's performance to within only 2 percentage point of clean test accuracy, while restoring 15.4% of lost accuracy.

Stage 3 solved the problem of working only with one dataset by merging the CICIDS2017, CSE-CICIDS2018, and TON IoT datasets into one corpus for training via a unified preprocessing workflow. It was established that an eight-feature representation common to all datasets exists, and is semantically equivalent. Training and testing of four neural network architectures using the consolidated two-million flow corpus took place under equal conditions. The best architecture, GRU, obtained an accuracy of 94.23% and F1 score of 0.8714, with all four passing the 91% threshold, indicating that flow representation of adequate dimensionality is capable of reliably detecting intrusions in heterogeneous environments. A leave-one-dataset-out evaluation further established that cross-domain F1 scores range from 0.45 to 0.69 when models are tested on entirely unseen network environments, confirming that distribution shift between enterprise and IoT traffic remains a fundamental challenge and providing a reproducible baseline for future domain adaptation research.

Bringing the results of the three stages together, it is possible to conclude that a comprehensive strategy should be adopted regarding the development and evaluation of IDS. Benchmark performance is achievable and essential, but insufficient on its own. Well-performing models on clean test data from the same network environment used during training might collapse under pressure or when deployed elsewhere. Adversarial stress-testing and cross-domain validation are critical elements for gaining a more realistic idea about the abilities of a system.

8.2 Future Scope

A number of areas come to mind as natural extensions of the research covered in this thesis. The most straightforward next step in the realm of adversarial attacks would involve incorporating constraints on the flow properties according to network protocol restrictions; currently, the attack framework does not check whether the perturbed features correspond to a feasible flow state, which means that more realistic attacks respecting TCP flag consistency, bounds on flow duration and packet counts, etc., could be devised.

With regards to defending against adversarial traffic, using adversarial training alongside input preprocessing techniques such as feature squeezing or denoising autoencoders may help bridge the gap between FNR on adversarial examples for the defended model (2.82%) and FNR on clean data (0.71%). Formal verification of neural network robustness through certified defenses is another interesting direction to explore, even though it is yet to be seen how they can be generalized to L_∞ norm.

For the cross-domain framework, scaling up to multi-class attack classification becomes the most urgent question. While binary classification served its purpose well here, operational IDSs will need to classify attacks into types in order to understand what is happening on the network level. This means constructing an attack taxonomy that is consistent across datasets, even if their labels use different terminology. Another way to enhance the framework would be the development of automated algorithms for aligning features, since the current selection process was based solely on manual inspection of eight specific features, while a principled approach may find other suitable feature sets.

Domain-adversarial training is a direct extension of the cross-domain framework presented here. The gradient reversal approach explicitly suppresses dataset-specific features during representation learning by training a feature extractor that simultaneously minimises attack classification loss and maximises domain classification loss. Given the leave-one-dataset-out baseline established in Stage 3, this technique is the most principled next step for improving cross-domain transfer without requiring additional labelled data from the target environment.

Another promising research avenue is that of online learning pipelines, which would allow updating a trained model incrementally when new data becomes available from a live environment. Lastly, the idea of scaling the cross-domain framework into the federated setting would solve the issue of privacy constraints that limit the scope of centralized training procedures across multiple datasets.

8.3 Social Impact

There are multiple levels of practical relevance for the contributions presented in this work. Firstly, the adversarial evaluation method developed in Stage 2 can be used by security engineers as a blueprint for robustness assessments performed prior to implementing an IDS solution. Understanding what perturbation budget leads to a false negative rate exceeding an operational threshold is a more informative measure of safety than a single clean-traffic AUC value. Adversarial training defence can be easily integrated into existing training pipelines and achieves significant robustness gains without significant degradation in performance on clean data. Cross-domain generalization provides a route to IDS implementation in which retraining is not required when environmental conditions change.

In addition, the systems considered here protect critical infrastructure which people use everyday — power generation plants, hospitals, water purification facilities, banking systems all require secure monitoring. An adversarially fragile IDS in such a scenario poses a threat — an experienced adversary familiar with the internal workings of the IDS system could potentially evade detection indefinitely.

BIBLIOGRAPHY

- [1] G. Karatas, O. Demir, and O. K. Sahingoz, "Deep learning in intrusion detection systems," in *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism*. Ankara, Turkey: IEEE, 2018, pp. 113–116.
- [2] J. Abraham and V. R. Bindu, "Intrusion detection and prevention in networks using machine learning and deep learning approaches: A review," in *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAEECA)*. IEEE, 2021, pp. 1–6.
- [3] R. Malhotra and S. Prasad, "Network intrusion detection for vulnerable networks using CICIDS2017: Preprocessing, SMOTE balancing, and comparative evaluation of random forest, XGBoost and neural network," in *Proceedings of the International Conference on Innovative Computing and Communication (ICICC 2026)*, New Delhi, India, 2026, to appear.
- [4] Y. Zhang, R. C. Muniyandi, and F. Qamar, "A review of deep learning applications in intrusion detection systems: Overcoming challenges in spatiotemporal feature extraction and data imbalance," *Applied Sciences*, vol. 15, no. 3, p. 1552, 2025.
- [5] E. C. P. Neto, S. Iqbal, S. Buffett, M. Sultana, and A. Taylor, "Deep learning for intrusion detection in emerging technologies: A comprehensive survey and new perspectives," *Artificial Intelligence Review*, vol. 58, no. 4, p. 340, 2025.
- [6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.
- [7] S. Srivastav, K. Guleria, and S. Sharma, "Machine learning based predictive model for intrusion detection," in *Proceedings of the International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*. IEEE, 2023, pp. 1–6.
- [8] N. Rana, H. Alshehri, M. A. Abdali, and W. A. Madkhali, "Optimized intrusion detection system for attack classification using machine learning and deep learning techniques," in *Proceedings of the 5th International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*. IEEE, 2024, pp. 1–7.
- [9] R. Varaprasad, A. P. Chakkaravarthy, and M. Veeresha, "A comprehensive analysis of intrusion detection system using machine learning and deep learning algorithms," in *Proceedings of the International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS)*. IEEE, 2024, pp. 1–6.
- [10] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021.
- [11] G. Shenbagalakshmi, K. Rithika, and T. Aiswerya, "An improved framework for network intrusion detection using machine learning," *Wireless Networks*, vol. 31, no. 2, pp. 1–15, 2025.

- [12] A. I. Getman, D. A. Rybolovev, and A. G. Nikolskaya, "Deep learning applications for intrusion detection in network traffic," *Programming and Computer Software*, vol. 50, no. 6, pp. 493–510, 2024.
- [13] M. Farhan *et al.*, "Network-based intrusion detection using deep learning technique," *Scientific Reports*, vol. 15, no. 1, p. 25550, 2025.
- [14] W. Dai, X. Li, W. Ji, and S. He, "Network intrusion detection method based on CNN-BiLSTM-Attention model," *IEEE Access*, vol. 12, pp. 53 099–53 111, 2024.
- [15] R. Chinnasamy, M. Subramanian, and S. Easwaramoorthy, "Deep learning-driven methods for network-based intrusion detection systems: A systematic review," *ICT Express*, vol. 11, no. 1, pp. 181–215, 2025.
- [16] B. Parakkal, T. Mazzuchi, and S. Sarkani, "Study of deep learning architectures for network intrusion detection," in *Proceedings of the ACM Southeast Conference*, 2025, pp. 297–298.
- [17] O. J. Mebawondu and B. Ajisafe-Badeji, "A multi-class intrusion detection model using an ensemble of deep learning techniques," in *Proceedings of the IEEE Nigeria Conference (NIGERCON)*. IEEE, 2024, pp. 1–5.
- [18] G. Nemiwal, D. Rani, and R. Kumar, "Transformer-based network intrusion detection: A multi-dataset analysis," in *Proceedings of the International Conference on Inventive Computing and Informatics (ICICI)*, 2025, pp. 966–972.
- [19] U. C. Akuthota and L. Bhargava, "Transformer based intrusion detection for IoT networks," *IEEE Internet of Things Journal*, vol. 12, pp. 6062–6067, 2025.
- [20] M. Cantone, C. Marrocco, and A. Bria, "Machine learning in network intrusion detection: A cross-dataset generalization study," *IEEE Access*, vol. 12, pp. 144 489–144 508, 2024.
- [21] K. Kostas, M. Just, and M. A. Lones, "Individual packet features are a risk to model generalization in ML-based intrusion detection," *IEEE Networking Letters*, vol. 7, no. 1, pp. 66–70, 2025.
- [22] V. Kurnala, H. S. Kanigolla, M. Manda, and A. Meda, "Cross-domain cybersecurity: Integrated intrusion detection framework," in *Proceedings of the Global Conference on Communications and Information Technologies (GCCIT)*. IEEE, 2024, pp. 1–5.
- [23] A. Aldaej, I. Ullah, T. A. Ahanger, and R. Atiqur, "Ensemble technique of intrusion detection for IoT-edge platform," vol. 14, 2024, p. 11703.
- [24] C. Westphal, S. Hailes, and M. Musolesi, "Feature selection for network intrusion detection," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025, pp. 1599–1610.
- [25] J. L. Hernandez-Ramos *et al.*, "Intrusion detection based on federated learning: A systematic review," *ACM Computing Surveys*, vol. 57, no. 12, pp. 1–38, 2025.
- [26] S. Agrawal, S. Sarkar, O. Aouedi, and G. Yenduri, "Federated learning for intrusion detection system: Concepts, challenges and future directions," *Computer Communications*, vol. 195, pp. 346–361, 2022.

- [27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.
- [28] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: From phenomena to black-box attacks using adversarial samples,” *arXiv preprint arXiv:1605.07277 [cs.CR]*, 2016.
- [29] K. He, D. D. Kim, and M. R. Asghar, “Adversarial machine learning for network intrusion detection systems: A comprehensive survey,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 538–566, 2023.
- [30] B. Alotaibi and M. A. Rassam, “Adversarial challenges in network intrusion detection systems: Research insights and future prospects,” *Future Internet*, vol. 15, no. 2, p. 62, 2023.
- [31] M. Catillo, A. D. Vecchio, A. Pecchia, and U. Villano, “A case study with CICIDS2017 on the robustness of machine learning against adversarial attacks in intrusion detection,” in *Proceedings of the 18th International Conference on Availability, Reliability and Security (ARES)*, Benevento, Italy, 2023, pp. 1–8.
- [32] Z. Awad, M. Zakaria, and M. El-Sayed, “An enhanced ensemble defense framework for boosting adversarial robustness of intrusion detection systems,” *Scientific Reports*, vol. 15, p. 14177, 2025.
- [33] J. Halvorsen, C. Izurieta, H. Cai, and A. Gebremedhin, “Applying generative machine learning to intrusion detection: A systematic mapping study and review,” *ACM Computing Surveys*, vol. 56, no. 10, pp. 1–33, 2024.
- [34] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, Madeira, Portugal, 2018, pp. 108–116.

Appendices

APPENDIX A

ADDITIONAL EXPERIMENTAL RESULTS

I.1 Per-Class Metrics Under Adversarial Attack

Table A.1 provides per-class precision, recall, and F1-score for the MLP under clean and FGSM-attacked ($\varepsilon = 0.1$) conditions.

Table A.1 Table-I.1: MLP Per-Class Metrics — Clean vs. FGSM ($\varepsilon = 0.1$)

Condition	Class	Precision	Recall	F1
Clean	Benign	0.9984	0.9678	0.9828
	Attack	0.9820	0.9929	0.9874
FGSM ($\varepsilon=0.1$)	Benign	0.9420	0.9048	0.9231
	Attack	0.7891	0.7740	0.7815

I.2 Detailed ε -Sweep Accuracy

Table A.2 Table-I.2: Accuracy vs. ε for All Models Under FGSM

Model	Accuracy at ε				
	0.01	0.05	0.10	0.20	0.30
MLP	0.876	0.819	0.768	0.747	0.743
RF	0.921	0.913	0.912	0.912	0.912
XGBoost	0.916	0.908	0.907	0.907	0.906

LIST OF PUBLICATIONS AND THEIR PROOFS

The following publication proofs (acceptance letters, conference certificates, indexing proofs, and payment receipts) are attached in this section.

1. List of Publications

1. Conference Paper C1:

R. Malhotra and S. Prasad, “Network Intrusion Detection for Vulnerable Networks using CICIDS2017: Preprocessing, SMOTE Balancing, and Comparative Evaluation of Random Forest, XGBoost and Neural Network,” *Proc. "9th International Conference on Innovative Computing and Communication (ICICC-2026)"*, Springer LNNS Series [Indexing: Scopus, DBLP, EI Compendex, SCImago], New Delhi, India, 06–07 February 2026.

2. Conference Paper C2:

S. Prasad and R. Malhotra, “Adversarial Evasion Attacks and Defense Strategies for ML-Based Network Intrusion Detection on CICIDS2017,” *Proc. "7th International Conference on Data Analytics & Management (ICDAM-2026)"*, Springer LNNS Series [Indexing: SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago], 12–14 June 2026.

3. Conference Paper C3:

S. Prasad and R. Malhotra, “A Cross-Domain Intrusion Detection System Using Unified Network Flow Representations and Deep Models,” *Proc. "7th International Conference on Data Analytics & Management (ICDAM-2026)"*, Springer LNNS Series [Indexing: SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago], 12–14 June 2026.

2. Paper Acceptance Proofs

2.1 Conference Paper C1 — ICICC 2026 Acceptance Letter



Suraj Prasad <spkalauni7789@gmail.com>

ICICC 2026: Paper Notification for Paper ID 951

3 messages

ICICC 2026 <icicc.ui@gmail.com>
To: Spkalauni <spkalauni7789@gmail.com>

Fri, Nov 21, 2025 at 7:30 PM

9th International Conference on Innovative Computing and Communication (ICICC-2026) – A Flagship Conference

Dear Author(s),

Greetings from - ICICC 2026!

We congratulate you that your paper with submission ID '951' and Paper Title '**Network Intrusion Detection for Vulnerable Networks using CICIDS2017: Preprocessing, SMOTE Balancing, and Comparative Evaluation of Random Forest, XGBoost and Neural Network**' has been accepted for publication in the **Proceedings of ICICC 2026- Springer LNNS Series** [Indexing: zbMATH, Scopus, DBLP, EI Compendex & SCImago - Approved]. This acceptance means your paper is among the top 15% of the papers received/reviewed.

To secure your participation in this prestigious event, we request you to complete your early-bird registration by 25th November 2025.

You are requested to do the registration as soon as possible and submit the following documents to icicc.ui@gmail.com at the earliest.

1. Final Camera-Ready Copy (CRC) as per the springer format- (See <https://icicc-conf.com/downloads>)
2. Copy of e-receipt of registration fees. (For Registration, see <https://icicc-conf.com/registrations>)
3. The final revised copy of your paper should also be uploaded via Microsoft CMT.
4. Do not create another ID, just upload the revised version on the existing Microsoft CMT ID only

The reviewers' comments are given at the bottom of this letter; please improve your paper as per the reviewers comments. **While preparing the final CRC manuscript, kindly check the following link under the download section tab named manuscript guidelines:**

<https://icicc-conf.com/registrations>

and it is suggested to cite the relevant latest papers matching the area of your current research paper.

The paper prior to submission should be checked for plagiarism from licensed plagiarism softwares like Turnitin/iAuthenticate etc. The similarity content should not exceed 15%.

Pay registration fees via online portal:

Pay via Account Transfer / QR Code (Indian Authors/ Outside India):

Pay via Bank Transfer

DECLARATION

We/I hereby certify that the work which is presented in the Major Project-II/Research Work entitled Towards Robust and Generalizable Machine Learning in fulfilment of the requirement for the award of the Degree of Bachelor/Master of Technology in Software Engineering and submitted to the Department of Software Engineering, Delhi Technological University, Delhi is an authentic record of my/our own, carried out during a period from 2024-26, under the supervision of Dr. Ruchika Malhotra.

The matter presented in this report/thesis has not been submitted by us/me for the award of any other degree of this or any other Institute/University. The work has been published/accepted/communicated in SCI/ SCI expanded/SSCI/Scopus indexed journal OR peer reviewed Scopus indexed conference with the following details:

Title of the Paper: Network Intrusion Detection for Vulnerable Networks using CICIDS2017: Preprocessing, SMOTE Balancing, and Comparative Evaluation of Random Forest, XGBoost and Neural Network

Author names (in sequence as per research paper): R. Malhotra and S. Prasad Kalauni

Name of Conference/Journal: 9th International Conference on Innovative Computing and Communication (ICICC-2026)

Conference Dates with venue (if applicable): 06th-07th February 2026, New Delhi, India

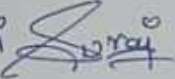
Have you registered for the conference (Yes/No)? Yes

Status of paper (Accepted/Published/Communicated): Accepted

Date of paper communication: 11 November 2025

Date of paper acceptance: 21 November 2025

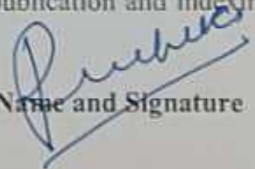
Date of paper publication: 2026 (Springer proceedings, post-conference)

24/5/26/25 Suraj Prasad Kalauni 
Student(s) Roll No., Name and Signature

SUPERVISOR CERTIFICATE

To the best of my knowledge, the above work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere. I, further certify that the publication and indexing information given by the students is correct.

Place: Delhi

Supervisor Name and Signature 

Date: 21/05/26

NOTE: PLEASE ENCLOSE RESEARCH PAPER
ACCEPTANCE/PUBLICATION/COMMUNICATION PROOF ALONG WITH SCOPUS
INDEXING PROOF

(Conference Website OR Science Direct in case of Journal Publication).

2.2 Conference Paper C2 — ICDAM 2026 Acceptance Letter (Paper ID 1532)



Suraj Prasad <spkalauni7789@gmail.com>

ICDAM 2026: Paper Notification for Paper ID 1532

11 messages

ICDAM 2026 <icdam.conf@gmail.com>
To: Spkalauni <spkalauni7789@gmail.com>

Sat, May 23, 2026 at 4:49 PM

7th International Conference on Data Analytics & Management (ICDAM-2026)!

Dear Author(s),

Greetings from ICDAM 2026!

We congratulate you that your paper with submission ID **1532** and Paper Title '**Adversarial Evasion Attacks and Defense Strategies for ML-Based Network Intrusion Detection on CICIDS2017**' has been accepted for publication in the Springer LNNS series [Approved]-[Indexing: SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago; All books published in the series are submitted for consideration in Web of Science]. This acceptance means your paper is among the top 20% of the papers received/reviewed. We urge you to complete your registration immediately to secure your spot at this highly anticipated event. **Kindly complete the registration by 25th May 2026. No further Extension.**

Please register as soon as possible and submit the following documents to icdam.conf@gmail.com as soon as possible.

1. Final Camera-Ready Copy (CRC) as per the springer format. (See <https://icdam-conf.com/downloads>)
2. Copy of e-receipt of registration fees. (For Registration, see <https://icdam-conf.com/registrations>)
3. The final revised copy of your paper must be mailed to the conference mail id.

Note : Standard Paper size – 10-12 pages. Over length of more than 12 pages, paper charges USD 20 per extra page

The reviewers comments are given at the bottom of this letter; please improve your paper as per the reviewers comments. While preparing the final CRC manuscript, kindly check the following google link of proceedings of the previous International Conference on Data Analytics and Management:

<https://link.springer.com/conference/icdam>

and it is suggested to cite the relevant latest papers matching the area of your current research paper.

The paper prior to submission should be checked for plagiarism from licensed plagiarism softwares like Turnitin/iAuthenticate etc. The similarity content should not exceed 15%, and AI content must not exceed 20%.

Pay registration fees via online portal:

DECLARATION

We/I hereby certify that the work which is presented in the Major Project-II/Research Work entitled Towards Robust and Generalizable Machine Learning based in fulfilment of the requirement for the award of the Degree of Bachelor/Master of Technology in Software Engineering and submitted to the Department of Software Engineering, Delhi Technological University, Delhi is an authentic record of my/our own, carried out during a period from 2024-26, under the supervision of Dr. Ruchi Ka Malhotra

The matter presented in this report/thesis has not been submitted by us/me for the award of any other degree of this or any other Institute/University. The work has been published/accepted/communicated in SCI/ SCI expanded/SSCI/Scopus indexed journal OR peer reviewed Scopus indexed conference with the following details:

Title of the Paper: Adversarial Evasion Attacks and Defense Strategies for ML-Based Network Intrusion Detection on CICIDS2017

Author names (in sequence as per research paper): S. Prasad Kalauni and R. Malhotra

Name of Conference/Journal: 7th International Conference on Data Analytics & Management (ICDAM-2026)

Conference Dates with venue (if applicable): 12-14 June 2026, London Metropolitan University, UK

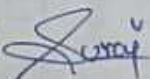
Have you registered for the conference (Yes/No)? : Yes

Status of paper (Accepted/Published/Communicated): Accepted

Date of paper communication: 19 May 2026

Date of paper acceptance: 23 May 2026

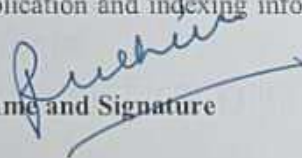
Date of paper publication: 2026 (Springer proceedings, post-conference)

24/SWE/03 Suraj Prasad Kalauni 
Student(s) Roll No., Name and Signature

SUPERVISOR CERTIFICATE

To the best of my knowledge, the above work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere. I, further certify that the publication and indexing information given by the students is correct.

Place: Delhi

Supervisor Name and Signature 

Date: 21/05/26

**NOTE: PLEASE ENCLOSE RESEARCH PAPER
ACCEPTANCE/PUBLICATION/COMMUNICATION PROOF ALONG WITH SCOPUS
INDEXING PROOF**

(Conference Website OR Science Direct in case of Journal Publication).

2.3 Conference Paper C3 — ICDAM 2026 Acceptance Letter (Paper ID 1585)



Suraj Prasad <spkalauni7789@gmail.com>

ICDAM 2026: Paper Notification for Paper ID 1585

2 messages

ICDAM 2026 <icdam.conf@gmail.com>
To: Spkalauni <spkalauni7789@gmail.com>

Sat, May 23, 2026 at 4:49 PM

7th International Conference on Data Analytics & Management (ICDAM-2026)!

Dear Author(s),

Greetings from ICDAM 2026!

We congratulate you that your paper with submission ID **1585** and Paper Title '**A Cross-Domain Intrusion Detection System Using Unified Network Flow Representations and Deep Models**' has been accepted for publication in the Springer LNNS series [Approved]-[Indexing: SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago; All books published in the series are submitted for consideration in Web of Science]. This acceptance means your paper is among the top 20% of the papers received/reviewed. We urge you to complete your registration immediately to secure your spot at this highly anticipated event. **Kindly complete the registration by 25th May 2026. No further Extension.**

Please register as soon as possible and submit the following documents to icdam.conf@gmail.com as soon as possible.

1. Final Camera-Ready Copy (CRC) as per the springer format. (See <https://icdam-conf.com/downloads>)
2. Copy of e-receipt of registration fees. (For Registration, see <https://icdam-conf.com/registrations>)
3. The final revised copy of your paper must be mailed to the conference mail id.

Note : Standard Paper size – 10-12 pages. Over length of more than 12 pages, paper charges USD 20 per extra page

The reviewers comments are given at the bottom of this letter; please improve your paper as per the reviewers comments. While preparing the final CRC manuscript, kindly check the following google link of proceedings of the previous International Conference on Data Analytics and Management:

<https://link.springer.com/conference/icdam>

and it is suggested to cite the relevant latest papers matching the area of your current research paper.

The paper prior to submission should be checked for plagiarism from licensed plagiarism softwares like Turnitin/iAuthenticate etc. The similarity content should not exceed 15%, and AI content must not exceed 20%.

Pay registration fees via online portal:

DECLARATION

We/I hereby certify that the work which is presented in the Major Project-II/Research Work entitled Towards Robust and Generalizable Machine Learning based Network Intrusion Detection Software Engineering in fulfilment of the requirement for the award of the Degree of Bachelor/Master of Technology in Software Engineering and submitted to the Department of Software Engineering, Delhi Technological University, Delhi is an authentic record of my/our own, carried out during a period from 2024-26, under the supervision of Dr. Ruchika Malhotra.

The matter presented in this report/thesis has not been submitted by us/me for the award of any other degree of this or any other Institute/University. The work has been published/accepted/communicated in SCI/ SCI expanded/SSCI/Scopus indexed journal OR peer reviewed Scopus indexed conference with the following methods for details:

Title of the Paper: A Cross-Domain Intrusion Detection System Using Unified Network Flow Representations and Deep Models

Author names (in sequence as per research paper): S. Prasad Kalauni and R. Malhotra

Name of Conference/Journal: 7th International Conference on Data Analytics & Management (ICDAM-2026)

Conference Dates with venue (if applicable): 12-14 June 2026, London Metropolitan University, UK

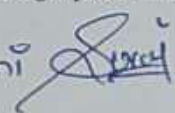
Have you registered for the conference (Yes/No)? : Yes

Status of paper (Accepted/Published/Communicated): Accepted

Date of paper communication: 19 May 2026

Date of paper acceptance: 23 May 2026

Date of paper publication: 2026 (Springer proceedings, post-conference)

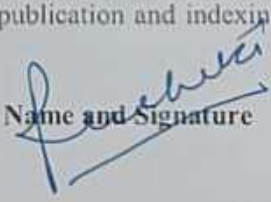
24/5/26/03 Suraj Prasad Kalauni 

Student(s) Roll No., Name and Signature

SUPERVISOR CERTIFICATE

To the best of my knowledge, the above work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere. I, further certify that the publication and indexing information given by the students is correct.

Place: Delhi

Supervisor Name and Signature 

Date: 21/05/26

**NOTE: PLEASE ENCLOSE RESEARCH PAPER
ACCEPTANCE/PUBLICATION/COMMUNICATION PROOF ALONG WITH SCOPUS
INDEXING PROOF**

(Conference Website OR Science Direct in case of Journal Publication).

3. Indexing Proof

3.1 ICICC 2026 — Springer LNNS Scopus Indexing Proof

09/05/2026, 16:50

ICICC | International Conference on Innovative Computing and Communication



ICICC
INTERNATIONAL CONFERENCE ON INNOVATIVE
COMPUTING AND COMMUNICATION

**10th
INTERNATIONAL
CONFERENCE ON
INNOVATIVE
COMPUTING AND
COMMUNICATION
(ICICC-2027)**

ORGANISED BY:
SHAHEED SUKHDEV
COLLEGE OF BUSINESS
STUDIES, UNIVERSITY
OF DELHI, NEW DELHI
IN ASSOCIATION WITH
UNIVERSITY OF
VALLADOLID SPAIN
5th-6th FEBRUARY
2027





ICICC 2026

The ninth edition of the International Conference on Innovative Computing and Communication (ICICC-2026) was successfully organized at Shaheed Sukhdev College of Business Studies, New Delhi, India, in association with the National Institute of Technology Patna, India, and University of Valladolid, Spain, on 06–07 February 2026.

The conference witnessed an overwhelming global response, receiving more than 2,500 research paper submissions from over 9,000 authors worldwide, reflecting the growing international interest in emerging computing and communication technologies. After a rigorous peer-review process, 475 high-quality papers were accepted, resulting in a competitive acceptance rate of approximately 18%.

All accepted papers were published in Springer's Lecture Notes in Networks and Systems, a Scopus-indexed publication series. The conference attracted over 1,000 participants including researchers, academicians, keynote speakers, industry experts, and delegates from around 40

Important Dates

Paper Submission

Deadline:

30th August 2026

Notification of first review:

25th October 2026

Submission of revised

3.2 ICDAM 2026 — Springer LNNS Scopus Indexing Proof

09/05/2026, 16:54

ICDAM



7th International Conference on Data Analytics & Management (ICDAM-2026)

Theme: Data Analytics with Computer Networks

Organized By: London Metropolitan University (UK)

In association with WSG University - Portalegre Polytechnic University - SGGW Management Institute

Date: 12-14 June 2026

Springer LNNS (Indexed in Scopus, EI, WoS)



About the Conference

The International Conference on Data Analytics and Management (ICDAM-2026) is organised with the objective of bringing together innovative scientists, professors, research scholars, students and industrial experts in the field of Computing and Communication to a common forum.

The primary goal of the conference is to promote the exchange of innovative scientific information between researchers, developers, engineers, students, and practitioners. Another goal is to promote the transformation of fundamental research into institutional and industrialized research and to convert applied exploration into real time application.

Overall, the conference will provide researchers and attendees with opportunities for national and international collaboration and networking among universities and institutions from India and abroad for promoting research.

<https://www.icdam-conf.com>

1/6

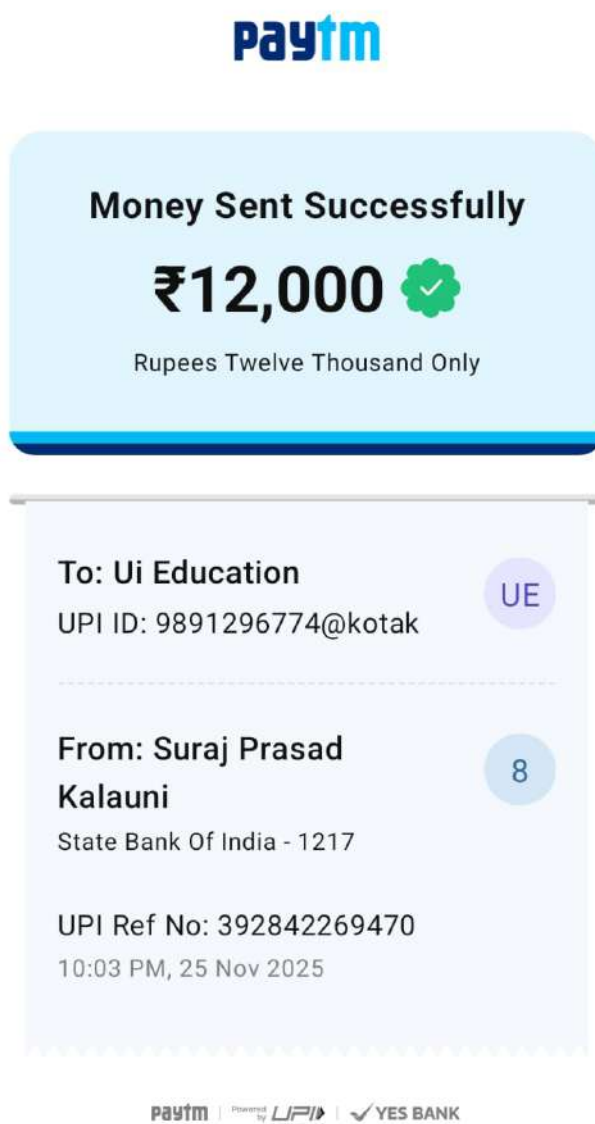
4. Conference Certificate

4.1 ICICC 2026 — Presenter Certificate (Suraj Prasad)

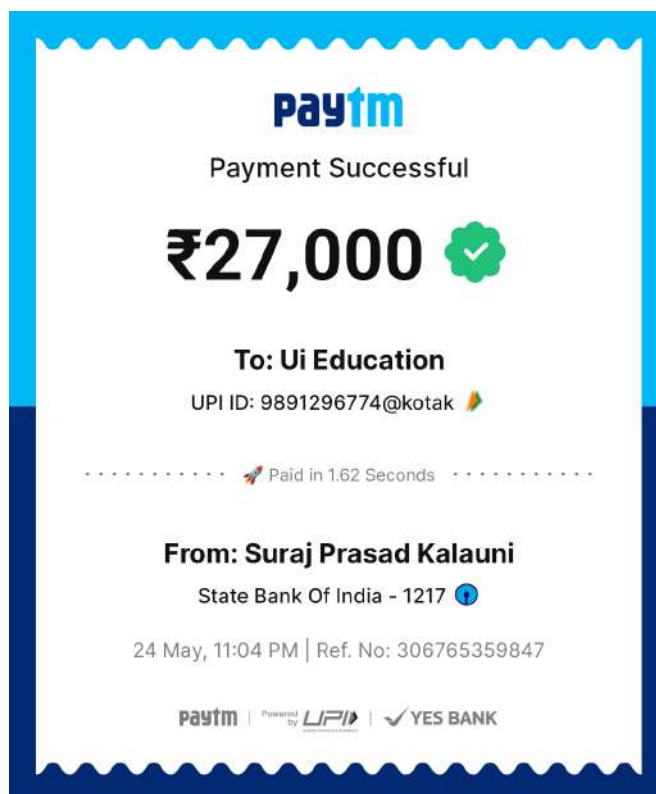


5. Conference Paper Registration Receipts

5.1 ICICC 2026 — Registration Receipt



5.2 ICDAM 2026 — Registration Receipt





DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daultpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis Towards Robust and Generalizable Machine Learning based
Network Intrusion Detection from Ensemble Baseline to Deep learning methods for
complex and
Total Pages _____ Name of the Scholar Suraj Prasad Kalavni
Supervisor (s) _____ Evolving Threat
Environments

- (1) Prof. Ruchika Malhotra
(2) _____
(3) _____

Department Software Engineering

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: Turnitin Similarity Index: 8%, Total Word Count: 19774

Date: 21/05/26

Candidate's Signature

Signature of Supervisor(s)

Rushika

Thesis

ORIGINALITY REPORT

9%

SIMILARITY INDEX

7%

INTERNET SOURCES

5%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Delhi Technological University Student Paper	2%
2	icdam-conf.com Internet Source	1%
3	easychair.org Internet Source	<1%
4	Submitted to DU Faculty Members Student Paper	<1%
5	arxiv.org Internet Source	<1%
6	aisa25.umpo.ma Internet Source	<1%
7	www.icdam-conf.com Internet Source	<1%
8	Submitted to Global Banking Training Student Paper	<1%
9	dspace.dtu.ac.in:8080 Internet Source	<1%
10	icicc-conf.com Internet Source	<1%
11	iot.uok.edu.in Internet Source	<1%
12	Wajdan Al Malwi, Fatima Asiri, Muhammad Shahbaz Khan. "Quantum-Enhanced Intrusion Detection Using Quantum Circuit	<1%

Feedback

0% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

- 
0 AI-generated only 0%
 Likely AI-generated text from a large-language model.
- 
0 AI-generated text that was AI-paraphrased 0%
 Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

AUTHOR'S PROFILE

Suraj Prasad Kalauni is currently pursuing the Master of Technology (M.Tech.) degree in Software Engineering from the Department of Software Engineering, Delhi Technological University (DTU), Delhi, India. His research work includes machine learning based network intrusion detection system (NIDS), adversarial robustness in cybersecurity, cross domain intrusion detection, and deep learning methods for intelligent threat analysis. His dissertation titled *“Towards Robust and Generalizable Machine Learning-Based Network Intrusion Detection: From Ensemble Baselines to Deep Learning Methods for Complex and Evolving Threat Environments”* includes ensemble learning techniques, adversarial robustness, and cross-domain deep learning methods for intrusion detection.



His research areas are Cybersecurity, Network Intrusion Detection Systems, Machine Learning, Deep Learning, Adversarial Machine Learning, Cross-Domain Generalization, Explainable AI, Transformer-based Architectures, and Intelligent Threat Detection Systems. Throughout his postgraduate studies, he have developed several methods regarding adversarial robustness analysis such as FGSM and PGD attacks, transfer-based attack (evasion), various ensemble techniques like Random Forest and XGBoost, CNN, LSTM, GRU and Transformer lightweight network models for network intrusion detection.

He has contributed to research publications including international conference papers in the domains of cybersecurity, adversarial machine learning, and intelligent intrusion detection. His research contributions include comparative evaluation of machine learning and deep learning-based IDS models, adversarial robustness analysis for CICIDS2017-based intrusion detection systems, and the development of a unified cross-domain intrusion detection framework using compact flow-level feature representations across “CICIDS2017, CSE-CICIDS2018, and TON-IoT” datasets.