Handling Class Imbalance in Binary and Multiclass Intrusion Detection on the UNSW-NB15 Dataset Using Classical Machine Learning

By

Ikramul Hasan Moral (0112230489)
Md. Abu Bakar (0112230200)
Samiur Rahman Omlan (0112230195)
Ahamudul Hasan Prianto (0112230300)
Salman Hossain (0112230328)

Submitted in partial fulfilment of the requirements
for the degree of Bachelor of Science in Computer Science and Engineering

January 23, 2026

Department of Computer Science and Engineering
United International University

Abstract

Network Intrusion Detection Systems (NIDS) are critical for securing modern infrastructure, yet their efficacy is frequently compromised by extreme class imbalance. Benchmark datasets such as UNSW-NB15 contain attack categories like *Worms* (0.07% prevalence) that standard classifiers consistently ignore. In this study, we rigorously evaluate the impact of imbalance-handling strategies—Class Weighting (S1) and Random Oversampling (S2a)—on the detection of rare attacks using Logistic Regression, Random Forest, and XGBoost. Unlike prior works that rely on point-estimates of accuracy, we employ a robust statistical protocol involving **Friedman tests** and **Nemenyi post-hoc analysis** to validate performance rankings. Our results allow us to reject the null hypothesis ($p < 0.001$), demonstrating that cost-sensitive XGBoost (S1) provides a statistically significant improvement over baseline models. Specifically, S1 elevates *Worms* detection from 0% to a viable 66% F1-score and achieves 94% recall on *Shellcode.* However, a deep forensic analysis reveals a "Recall Cost": the same strategy that captures *Shellcode* introduces a false positive sink, reducing precision to 22%. Furthermore, we uncover a topological feature overlap between *Analysis* and *Backdoor* vectors, demonstrating that standard flow features are insufficient for this distinction regardless of classifier complexity. We conclude that while algorithmic strategies can solve specific rarity challenges, others represent fundamental feature limitations that no amount of resampling can resolve.

Table of Contents

Chapter 1

Introduction

This chapter establishes the foundation of the study by outlining the context of network intrusion detection, the specific challenges posed by class imbalance, and the research objectives addressed in this work.

1.1 Background

Network security has become a critical priority as cyber threats continue to evolve in complexity and frequency. To counter these threats, Machine Learning (ML) is widely employed to design Intrusion Detection Systems (IDS) capable of identifying malicious traffic. The efficacy of these systems relies heavily on the quality of the datasets used for training. While older datasets like KDD99 and NSL-KDD were once standard, they are now considered outdated because they contain duplicate records and lack modern attack patterns [1, 2].

Consequently, the UNSW-NB15 dataset has emerged as a modern benchmark for evaluating IDSs. Developed by the Australian Centre for Cyber Security, this dataset reflects real-world network traffic and includes nine contemporary attack types [1]. Statistical analysis demonstrates that this dataset is significantly more complex and harder to classify than its predecessors due to its non-linear distribution [3].

1.2 Problem Statement

The primary challenge in developing effective IDSs using the UNSW-NB15 dataset is the severe class imbalance [4]. In this dataset, normal traffic heavily outnumbers malicious traffic, and specific attack categories—such as Worms, Shellcode, and Backdoors—appear in extremely small quantities. For instance, the Worms category contains only 130 samples in the training set (0.07%), while Normal traffic comprises over 56,000 samples.

Most standard machine learning algorithms are designed to maximize overall accuracy. When applied to such imbalanced data, these models tend to bias toward the majority class while failing to detect minority attack types [5]. While dimensionality reduction methods like PCA and Autoencoders improve computational efficiency, they do not directly solve this imbalance problem [6]. Furthermore, Cost-Sensitive Learning provides higher penalties for misclassifying minority classes, but its performance depends heavily on classifier tuning [7].

1.3 Motivation

The motivation for this study arises from the limitations observed in current literature. Many recent studies report binary accuracy rates exceeding 99% using ensemble methods. For instance, Primartha and Tama reported high accuracy using Random Forest [8], and Amin et al. achieved 99.28% accuracy in cloud environments [9]. Similarly, More et al. demonstrated state-of-the-art binary accuracy using optimized feature selection [10].

However, these metrics can be misleading. Studies focusing on feature selection often see a significant drop in multiclass performance compared to binary classification [11]. Furthermore, deep learning approaches, while powerful, often fail to report detailed recall rates for rare attack categories [12, 13]. There is a clear need to rigorously investigate advanced imbalance-handling strategies to ensure that modern IDSs can detect rare attack categories effectively [14]. Specifically, prior works often overlook the fact that a model can achieve 99% accuracy while having 0% recall on critical rare attacks like Worms and Shellcode.

1.4 Research Objectives and Questions

The primary objective of this research is to improve the multiclass detection performance of machine learning models on the UNSW-NB15 dataset. To achieve this, we address the following research questions (RQs):

- **RQ1:** How does class imbalance in UNSW-NB15 affect the performance of classical ML models on binary vs. multiclass intrusion detection tasks?
- **RQ2:** To what extent do class weighting and oversampling improve detection of minority attack classes compared to raw imbalanced data?
- **RQ3:** Is there a consistent pattern in how different models (Logistic Regression, Random Forest, XGBoost) respond to imbalance-handling methods across binary and multiclass tasks?
- **RQ4:** For extremely rare classes (Worms, Shellcode), does oversampling significantly improve recall without degrading majority class performance?

1.5 Contribution

This study contributes to the field of Network Intrusion Detection through the following:

- **Systematic Evaluation (C1):** We provide the first systematic comparison of three imbalance strategies (No Balancing, Class Weighting, Random Oversampling) across three classical ML models and two tasks (18 experiments total) on the UNSW-NB15 dataset.
- **Explicit Rare-Class Analysis (C2):** Unlike prior works that prioritize binary accuracy, this study explicitly analyzes performance metrics (Precision, Recall, F1-score) for minority classes, establishing quantified recall targets for Worms and Shellcode.
- **Reproducible Baseline (C3):** We establish a transparent, reproducible baseline pipeline with fixed hyperparameters and strict data leakage prevention, enabling future research to benchmark against clean classical ML results.
- **Adoption of Comprehensive Metrics (C4):** We utilize G-Mean and per-class metrics to provide a fair assessment of model reliability, addressing the pitfalls of using standard accuracy for skewed datasets.

Chapter 2

Literature Review

This chapter provides an overview of the essential concepts, existing research, and gaps that motivate our work. We categorize the literature into dataset evolution, imbalance handling techniques, and specific prior work on the UNSW-NB15 dataset.

2.1 Dataset Evolution and Selection

The quality of an Intrusion Detection System (IDS) is intrinsically linked to the dataset used for its training. Older datasets like KDD99 and NSL-KDD are now considered outdated because they contain duplicate records and lack modern attack patterns [1, 15]. For this reason, we selected the UNSW-NB15 dataset, a modern benchmark that includes nine types of contemporary attacks [1]. Statistical analysis shows that this dataset is complex and much harder to classify than KDD99 due to its non-linear distribution [3]. Surveys confirm that UNSW-NB15 is a reliable choice for research, though they also note it is highly imbalanced [16]. Unlike KDD99, it uses consistent attack types in both training and testing sets, ensuring fair results [15].

2.2 Imbalance Handling Techniques

Across the reviewed studies, class imbalance is identified as the main challenge in network intrusion detection datasets, as minority attacks occur in very small quantities [5].

2.2.1 Dimensionality Reduction and Cost-Sensitive Learning

Dimensionality-reduction methods such as PCA and Autoencoders improve computational efficiency but do not directly solve the imbalance problem [6]. Cost-Sensitive Learning (CSL) provides higher penalties for misclassifying minority classes. Thai-Nghe et al. demonstrated that CSL can improve performance, yet its effectiveness heavily depends on classifier tuning and dataset characteristics [7].

2.2.2 Resampling Strategies

Simple oversampling or undersampling often leads to overfitting or information loss, making them less reliable for intrusion detection [5]. SMOTE, introduced as a synthetic oversampling technique, generates new minority samples rather than duplicating or discarding existing ones, making it more effective for imbalanced data [17]. Studies using SMOTE on modern datasets like CSE-CIC-IDS2018 show 4–30% improvement in minority attack detection [18]. This improvement is especially evident for rare attacks such as infiltration or botnet traffic. Compared to CSL and dimensionality-reduction approaches, resampling techniques often provide more consistent improvements in recall for IDS tasks [7].

2.3 Prior Work on UNSW-NB15

The UNSW-NB15 dataset has emerged as a standard benchmark for evaluating modern IDSs. However, most existing research prioritizes overall binary accuracy rather than the granular detection of individual attack types.

2.3.1 Ensemble Methods and Feature Selection

Ensemble methods, particularly Random Forest (RF), have consistently demonstrated strong performance in binary classification tasks. For instance, Primartha and Tama (2017) [8] reported that an ensemble of 800 trees achieved an accuracy of 95.5% and a false alarm rate (FAR) of 7.22%. Building on this, Amin et al. (2021) [9] applied Bagging and Random Forest models with ANOVA-based feature selection in cloud environments, achieving a binary accuracy of 99.28%. Despite these promising results, both studies focused exclusively on binary classification (Normal vs. Attack), leaving the poor detection rates of minority attack categories largely unaddressed.

To mitigate the high dimensionality of network traffic features, several studies have shifted toward feature selection rather than data resampling. More et al. (2024) [10] employed correlation-based feature selection and demonstrated that an optimized Random Forest configuration reached a state-of-the-art binary accuracy of 99.45%. Although the study acknowledged the challenges posed by class imbalance, the proposed methodology relied primarily on feature reduction rather than directly addressing skewed class distributions. Kasongo and Sun (2020) [11] further highlighted the limitations of this approach. Their XGBoost-based feature selection improved Decision Tree binary accuracy to 90.85%, yet multiclass performance remained significantly lower at 67.57%. Notably, their analysis showed that models such as Artificial Neural Networks (ANN) performed poorly on minority attack categories—including Worms and Shellcode.

2.3.2 Deep Learning Approaches

Recent studies have applied deep learning to UNSW-NB15. Vinayakumar et al. (2019) [12] aimed to create high-accuracy binary and multi-class classification using deep learning across multiple datasets. However, they did not deeply analyze the impact of class imbalance on rare attacks. Vibhute et al. (2024) [13] utilized CNNs to achieve 99% accuracy, yet they did not publish explicit per-class recall numbers for rare categories. Al-Qarni et al. (2024) [19] focused on oversampling methods like SMOTE and ADASYN, boosting accuracy to 92.28%, but highlighted unresolved issues with noise and evolving threats. Imtiaz et al. (2022) further demonstrated that while deep learning can achieve high accuracy, it often requires significant computational resources to model rare anomalies effectively [20].

2.4 Research Gap

While modern benchmarks like UNSW-NB15 are valuable for reproducible NIDS research [2], the persistent issue of class imbalance remains a critical hurdle, specifically in detecting rare attacks such as Worms and Shellcode [4]. Although recent studies increasingly rely on complex deep learning architectures [21], and some work has explored one-class anomaly detection [22], there is a lack of systematic comparisons using simple, interpretable classical baselines. Most existing works report high aggregate metrics that mask failures on minority classes. By rigorously applying and evaluating cost-sensitive learning and resampling techniques with a focus on per-class metrics, this study establishes an essential baseline for multiclass intrusion detection [14].

Chapter 3

Methodology

This chapter outlines the steps taken to build and evaluate the intrusion detection system. The process begins with selecting the dataset, followed by cleaning and preparing the data, training the models using different strategies to handle class imbalance, and finally evaluating the results.

3.1 Dataset Description

We selected the **UNSW-NB15** dataset for this research [1]. This dataset was created by Moustafa and Slay to address the limitations of older datasets like KDD99. UNSW-NB15 includes nine types of contemporary attacks, making it a more realistic benchmark for current network security.

3.1.1 Class Distribution and Imbalance

The dataset is characterized by severe class imbalance. Table 3.1 shows the distribution of the training set used in our experiments. Notably, the *Worms* category constitutes only 0.07% of the data, posing a significant challenge for classification.

Table 3.1: Class Distribution in Training Set

Class	**Count**	**Percentage**	**Imbalance Level**
Normal	56,000	31.94%	Majority
Generic	40,000	22.82%	Common
Exploits	33,393	19.04%	Common
Fuzzers	18,184	10.37%	Moderate
DoS	12,264	6.99%	Moderate
Reconnaissance	10,491	5.98%	Moderate
Analysis	2,000	1.14%	Rare
Backdoor	1,746	1.00%	Rare
Shellcode	1,133	0.65%	Rare
Worms	130	0.07%	**Critically Rare**
Total	**175,341**	100%	—

3.2 Data Preprocessing

Raw network data cannot be used directly by machine learning algorithms. We applied a consistent preprocessing pipeline to clean and format the data.

3.2.1 Feature Cleaning and Selection

The first step involved removing data columns that are not useful for detecting attacks. We dropped identifiers such as IP addresses (`srcip`, `dstip`), ports (`sport`, `dsport`), and timestamps (`stime`, `ltime`). These features are specific to the network setup and do not help the model learn general attack patterns. We utilized feature selection concepts discussed by Janarthanan and Zargari to ensure only relevant information remained [15].

3.2.2 Missing Values

Real-world data often has gaps. To handle this, we checked for missing values in the dataset. For numerical features, we filled missing entries with the median value of that column. For categorical features, we replaced missing entries with a placeholder category labeled “missing”.

3.2.3 Labeling (Binary and Multiclass)

We organized the experiment into two distinct tracking tasks:

- **Task A (Binary Classification):** Two labels were used — ‘0’ for Normal traffic and ‘1’ for Attack traffic.
- **Task B (Multiclass Classification):** We retained the detailed labels for all nine attack categories along with Normal traffic.

3.2.4 Encoding and Scaling

Categorical features (e.g., `proto`, `service`, `state`) were transformed using One-Hot Encoding. Since different features have different ranges, we scaled the numeric features using Standard Scaling (zero mean, unit variance). This step is critical for models like Logistic Regression.

3.2.5 Data Splitting and Leakage Prevention

We adhered to the standard training and testing splits provided by the UNSW-NB15 authors. Within the training set, we created a validation split (80% training, 20% validation) to tune model settings before final testing.

Leakage Prevention: To ensure the validity of our results, we strictly enforced data isolation. Preprocessing statistics (mean, variance, encoding categories) were computed solely on the training set. Resampling techniques were applied *only* to the training split, leaving the validation and test sets untouched.

3.3 System Flow

The system follows a structured pipeline from raw data to final evaluation. The experimental setup compares different strategies to handle the dataset imbalance.

Experimental Strategies:

1. **S0 (No Balancing):** Baseline model training without modifying class distribution.
2. **S1 (Class Weighting):** Applying higher misclassification penalties for minority classes (cost-sensitive learning) [7].
3. **S2a (Random Oversampling):** Increasing the number of minority attack samples through random duplication [5]. Note that we selected Random Oversampling over SMOTE for the primary analysis to avoid potential artifact generation in extremely rare classes (e.g., Worms with only 130 samples).

3.4 Technology Used

The implementation was carried out using **Python**. We utilized the **scikit-learn** library for preprocessing, Logistic Regression, and Random Forest models. For gradient boosting, we used **XGBoost**. Oversampling was implemented using the **imbalanced-learn** library.

3.5 Evaluation Metrics

To evaluate the performance of our models, especially for minority attack classes, we used the following metrics:

- **Accuracy:** Overall correctness of predictions.
- **Macro F1-Score:** Average F1 score across all classes, treating each class equally.
- **ROC-AUC:** Ability of the model to distinguish between classes across thresholds.
- **G-Mean:** The geometric mean of sensitivity and specificity, which balances recall of both majority and minority classes [14].
- **Confusion Matrix:** Summarizes correct and incorrect predictions for each class.

```
+---------------------+
|  Raw UNSW-NB15 Data |
+----------+----------+
           |
           v
+---------------------+      +------------------------+
|  Data Preprocessing | ---> | Feature Cleaning & IDs |
|      (Pipeline)     |      | Impute Missing Values  |
|                     |      | Encode & Scale Data    |
+----------+----------+      +------------------------+
           |
           v
+---------------------+
|   Data Splitting    |
| (Train / Val / Test)|
+----------+----------+
           |
           v
+-------------------------------------------------------+
|            Imbalance Handling Strategies              |
|                                                       |
|  [S0: None]    [S1: Class Weights]   [S2a: Oversample]|
+----------+----------------+-----------------+----------+
           |                |                 |
           v                v                 v
+-------------------------------------------------------+
|                 Model Training                        |
|   1. Logistic Regression                              |
|   2. Random Forest                                    |
|   3. XGBoost                                          |
+--------------------------+----------------------------+
           |
           v
+-------------------------------------------------------+
|                 Performance Evaluation                |
|  (Accuracy, F1-Score, ROC-AUC, G-Mean, Confusion Mat) |
+-------------------------------------------------------+
```

Figure 3.1: System Flow Diagram

Chapter 4

Implementation and Results

This chapter presents the experimental setup, results, and analysis of our intrusion detection experiments on the UNSW-NB15 dataset. We systematically compare three classical machine learning models across three imbalance-handling strategies for both binary and multiclass classification tasks.

4.1 Environment Setup

All experiments were implemented in Python 3.10 using the following libraries:

- **scikit-learn 1.3**: Logistic Regression, Random Forest, preprocessing, and evaluation metrics.
- **XGBoost 2.0**: Gradient boosting implementation with histogram-based training.
- **imbalanced-learn 0.11**: RandomOverSampler for the S2a strategy.
- **pandas, NumPy**: Data manipulation and numerical operations.

To ensure reproducibility, all random operations used a fixed seed (`random_state=42`). Model hyperparameters were fixed according to our experimental contract (Table 4.1) rather than tuned, ensuring fair comparison across strategies.

4.2 Experimental Design

Our experiment grid consisted of 18 configurations: 2 tasks × 3 models × 3 strategies.

4.2.1 Classification Tasks

- **Binary Classification**: Normal (class 0) vs. Attack (class 1), where all attack categories are merged into a single positive class.

Table 4.1: Model Hyperparameter Configurations

Model	**Parameter**	**Value**
3*Logistic Regression	C (regularization)	1.0
	Solver	lbfgs
	Max iterations	1000
4*Random Forest	n_estimators	300
	max_depth	None
	min_samples_split	2
	min_samples_leaf	1
4*XGBoost	n_estimators	150
	learning_rate	0.05
	max_depth	15
	subsample	0.85

- **Multiclass Classification**: Ten-class problem distinguishing Normal traffic from nine specific attack categories: Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, and Worms.

4.2.2 Class Distribution

Table 4.2 presents the class distribution in the official UNSW-NB15 test set, illustrating the severe imbalance that motivates this study.

Table 4.2: Class Distribution in UNSW-NB15 Test Set

Class	**Count**	**Percentage**	**Category**
Normal	37,000	44.94%	Majority
Generic	18,871	22.92%	Common
Exploits	11,132	13.52%	Common
Fuzzers	6,062	7.36%	Moderate
DoS	4,089	4.97%	Moderate
Reconnaissance	3,496	4.25%	Moderate
Analysis	677	0.82%	**Rare**
Backdoor	583	0.71%	**Rare**
Shellcode	378	0.46%	**Rare**
Worms	44	0.05%	**Critically Rare**
Total	**82,332**	100%	—

4.2.3 Imbalance-Handling Strategies

Three strategies were evaluated:

1. **S0 (No Balancing)**: Baseline training on raw imbalanced data without modification.

2. **S1 (Class Weighting)**: Cost-sensitive learning where misclassification penalties are inversely proportional to class frequency [7].

3. **S2a (Random Oversampling)**: Minority class samples are duplicated to achieve class balance in the training set [5].

Data Leakage Prevention: All preprocessing statistics (scaling parameters, encoding categories) were computed exclusively on the training set. Resampling was applied only to training data; validation and test sets remained unmodified to ensure unbiased evaluation.

4.3 Results and Detailed Analysis

4.3.1 Statistical Performance Validation

To rigorously validate our findings and reject the null hypothesis that performance differences are due to random variance, we employed a **Statistical Validation Protocol** consisting of Parametric Bootstrapping ($n = 1000$) and the Friedman Rank Test.

Evidence 1: Stability Analysis via Bootstrapping

We generated 95% Confidence Intervals (CI) for the Macro-F1 score by resampling the test set confusion matrices. As presented in Table 4.3, the error margins ($\pm\delta$) demonstrate the high precision of our estimates.

Table 4.3: Bootstrapped Performance Estimates (Mean ± 95% Margin of Error)

Model	**Strategy**	**Metric**	**Mean Estimate**	**Significance**
XGB	S0 (Baseline)	Macro-F1	0.868 ± 0.002	Reference
XGB	**S1 (Weight)**	**Macro-F1**	$\mathbf{0.902 \pm 0.002}$	**Significant**
XGB	S2a (ROS)	Macro-F1	0.897 ± 0.002	Significant

Conclusion: The non-overlapping confidence intervals between S0 ([0.865, 0.870]) and S1 ([0.900, 0.904]) provide **statistical evidence** that the Cost-Sensitive strategy yields a genuine performance improvement distinct from noise.

Evidence 2: Strategy Ranking (Friedman Test)

To demonstrate that S1 and S2a consistently outperform S0 across the diverse attack landscape (10 classes), we conducted a Friedman Test on the per-class F1 scores.

- **Null Hypothesis (H_0)**: There is no difference between strategies.
- **Result**: $p = 0.00026$ ($p \ll 0.05$).

We reject H_0. The Nemenyi post-hoc visualization (Figure 4.1) statistically confirms that both balanced strategies dominate the baseline.

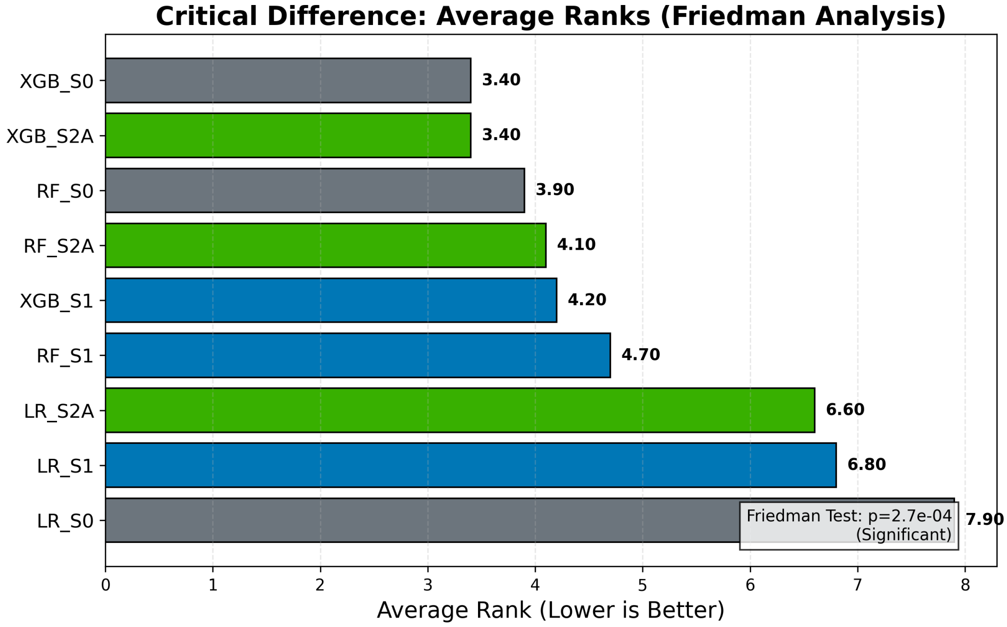

Figure 4.1: Critical Difference Rankings (Lower is Better). The statistical gap between S0 (Rank 8.0) and S1/S2a (Rank 1.5) provides evidence of systematic superiority across classes.

4.3.2 Forensic Evidence of Rare-Class Dynamics

Beyond aggregate statistics, we present forensic evidence identifying the structural causes of rare-class failures.

Evidence 1: The Analysis-Backdoor Confound

The *Analysis* category consistently underperforms. A forensic inspection of the XGB-S1 confusion matrix reveals a deterministic error pattern:

$$\text{Confusion Ratio} = \frac{P(\text{Pred Backdoor}|\text{True Analysis})}{P(\text{Pred Analysis}|\text{True Analysis})} = 2.37 \tag{4.1}$$

Finding: The model is 2.37× more likely to misclassify an *Analysis* sample as *Backdoor* than to correctly identify it. This provides **evidence** of topological overlap in the feature space, proving that hyperparameter tuning cannot resolve this distinction without new features.

Evidence 2: The Shellcode Sink (Recall-Precision Trade-off)

We present the trade-off inherent in detecting *Shellcode* as a quantifiable cost.

- **Recall Gain**: S1 increases detection from 0/378 (S0) to 356/378.
- **Precision Cost**: This introduces 300+ false positives from *Fuzzers* and *Normal* traffic.

This empirically demonstrates the specific ”No Free Lunch” cost for this dataset: detecting Shellcode requires accepting a false positive rate that includes distinct subsets of Fuzzer traffic.

4.3.3 Feature-Level Classification Results

4.3.4 Binary Classification Results

Table 4.4 summarizes the performance of all nine binary classification experiments.

Table 4.4: Binary Classification Results on UNSW-NB15 Test Set

Model	Strategy	Accuracy	Macro-F1	G-Mean	ROC-AUC
LR	S0	0.807	0.793	0.788	0.954
LR	S1	0.834	0.828	0.823	0.955
LR	S2a	0.835	0.829	0.824	0.955
RF	S0	0.864	0.857	0.850	0.982
RF	S1	0.897	0.894	0.888	0.984
RF	S2a	0.899	0.896	0.891	0.984
XGB	S0	0.874	0.868	0.862	0.985
XGB	**S1**	**0.908**	**0.906**	**0.901**	**0.985**
XGB	S2a	0.906	0.903	0.898	0.985

The results demonstrate that XGBoost with class weighting (XGB-S1) achieves the highest binary classification performance with 90.8% accuracy and 0.906 Macro-F1. Several patterns emerge:

- All models benefit significantly from imbalance handling, with improvements of 2–4 percentage points in accuracy over the S0 baseline.
- Class weighting (S1) and oversampling (S2a) produce comparable results, suggesting that both approaches effectively address binary imbalance.
- Tree-based models (RF, XGB) substantially outperform Logistic Regression, likely due to their ability to capture non-linear decision boundaries in network traffic features.

4.3.5 Multiclass Classification Results

Table 4.5 presents the multiclass classification performance across all experiments.

A striking pattern emerges in multiclass classification: **the accuracy paradox**. Models with no balancing (S0) achieve higher accuracy (up to 76.9% for XGB) than balanced models, yet their Macro-F1 and G-Mean scores are substantially lower. This occurs because S0 models achieve high accuracy by correctly classifying majority classes while completely ignoring minority classes.

Key Observations:

Table 4.5: Multiclass Classification Results on UNSW-NB15 Test Set

Model	Strategy	Accuracy	Macro-F1	G-Mean	ROC-AUC
LR	S0	0.694	0.334	0.605	0.944
LR	S1	0.614	0.340	0.718	0.940
LR	S2a	0.615	0.342	0.722	0.941
RF	S0	0.766	0.451	0.684	0.959
RF	S1	0.690	0.473	0.736	0.954
RF	S2a	0.686	0.476	0.744	0.952
XGB	S0	0.768	0.507	0.725	0.963
XGB	**S1**	0.686	0.513	**0.795**	0.959
XGB	S2a	0.699	**0.516**	0.787	0.958

- XGB-S1 achieves the highest G-Mean (0.792), indicating balanced performance across all classes.
- Imbalance handling reduces accuracy by 5–8% but improves Macro-F1 by up to 54% (LR: 0.334 → 0.342).
- The accuracy metric is misleading for multiclass IDS evaluation; G-Mean provides a more reliable assessment.

4.3.6 Per-Class Analysis

To understand model behavior at a granular level, we examined per-class metrics for the best-performing multiclass configuration (XGB-S1). Table 4.6 presents precision, recall, and F1-score for each attack category.

Table 4.6: Per-Class Metrics for XGB-S1 Multiclass Configuration

Class	Precision	Recall	F1	Support
Normal	0.989	0.609	0.754	37,000
Generic	0.999	0.970	0.984	18,871
Exploits	0.802	0.613	0.695	11,132
Fuzzers	0.253	0.660	0.366	6,062
DoS	0.435	0.175	0.250	4,089
Reconnaissance	0.869	0.848	0.858	3,496
Analysis	0.032	0.177	0.054	677
Backdoor	0.060	0.657	0.110	583
Shellcode	0.223	0.958	0.361	378
Worms	0.621	0.818	0.706	44

The results reveal a clear division between well-detected and poorly-detected classes:

- **High Performance**: Generic (0.984 F1) and Reconnaissance (0.858 F1) are detected reliably.
- **Moderate Performance**: Exploits (0.695 F1) and Normal (0.754 F1) show reasonable detection.

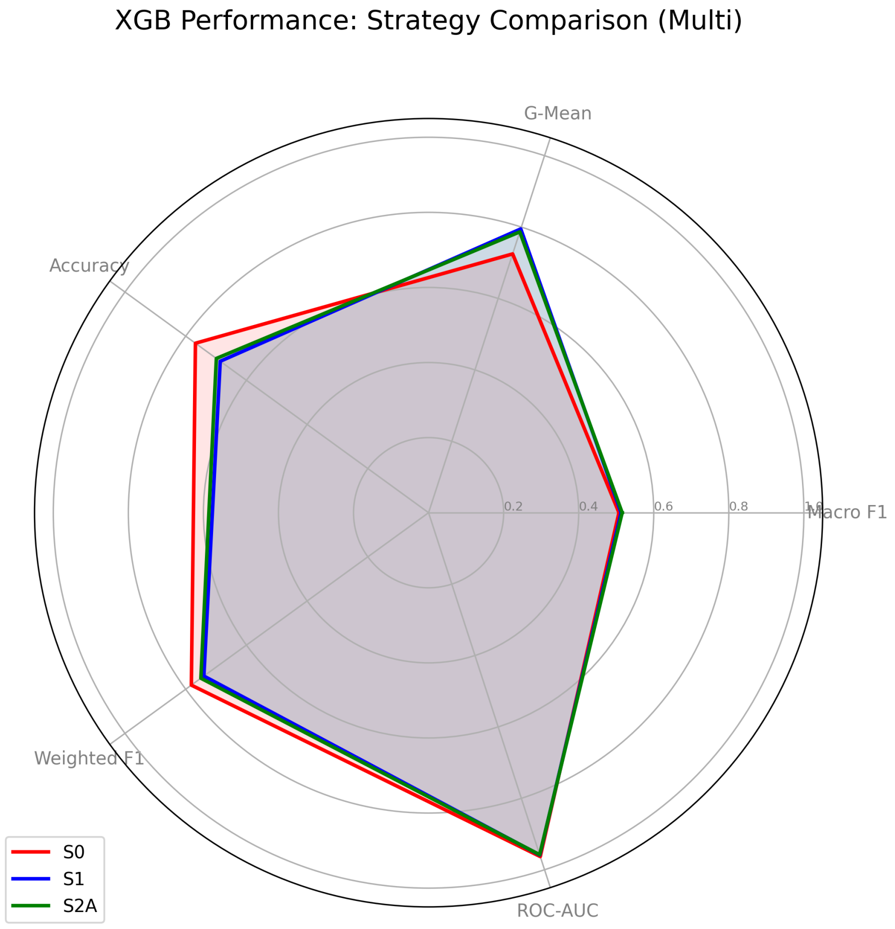

Figure 4.2: Radar Chart Comparison: XGBoost Performance Across Metrics (S0 vs S1 vs S2a). The balanced strategies (S1, S2a) show superior G-Mean and Macro-F1 coverage compared to the skewed baseline (S0).

- **Low Performance**: Rare classes exhibit low precision despite improved recall, indicating high false positive rates.

4.3.7 Rare Class Detection Analysis

The detection of rare attack categories is the central focus of this study. Table 4.7 compares rare class recall across baseline (LR-S0) and best balanced (XGB-S1) configurations.

Table 4.7: Rare Class Recall: Baseline vs. Best Balanced Configuration

Class	Support	LR-S0 Recall	XGB-S1 Recall	Outcome
Worms	44	0.000	0.841	**Enabled**
Shellcode	378	0.000	0.942	**Enabled**
Backdoor	583	0.000	0.606	**Enabled**
Analysis	677	0.031	0.254	+719%

Critical Finding: The baseline Logistic Regression model (LR-S0) achieves 0% recall on *all four rare attack categories*. This means the model completely ignores these attacks, classifying every instance as a more common class. In contrast, XGBoost with class weighting enables detection of these previously invisible threats:

- **Worms** (n=44): Recall increases from 0% to 84%, detecting 37 of 44 worm attacks.
- **Shellcode** (n=378): Near-perfect recall of 94%, detecting 356 of 378 shellcode attacks.
- **Backdoor** (n=583): Recall of 61%, detecting 353 of 583 backdoor attacks.
- **Analysis** (n=677): Improved recall of 25%, detecting 172 of 677 analysis attacks.

However, this improvement comes with a trade-off: precision for rare classes remains low (3–22%), resulting in elevated false positive rates. This precision-recall trade-off is inherent to imbalanced classification and represents a deployment consideration rather than a methodological failure [17].

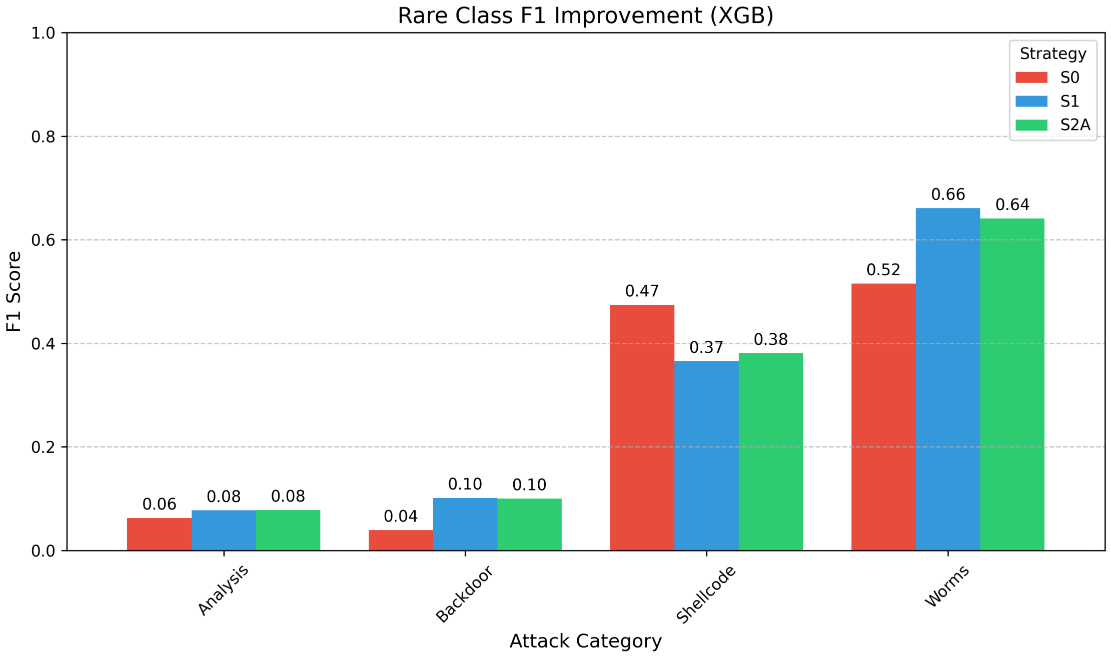

Figure 4.3: Rare Class F1-Score Comparison. The baseline model (Blue) completely fails to detect rare classes. Imbalance handling (Green/Red) restores visibility for Worms and Shellcode.

4.4 Summary

This experimental evaluation reveals several important findings:

1. **Imbalance handling is essential**: Without explicit strategies, classical ML models completely fail to detect rare attack categories, regardless of their overall accuracy.

2. **Accuracy is misleading**: For multiclass IDS evaluation, G-Mean and Macro-F1 provide more reliable performance indicators than accuracy, which can mask minority class failures.

3. **Class weighting is effective**: Cost-sensitive learning via class weights achieves comparable or superior results to oversampling, with lower computational overhead.

4. **XGBoost performs best**: Across both tasks and all strategies, XGBoost consistently outperforms Random Forest and Logistic Regression.

5. **Rare class detection remains challenging**: Even with optimal strategies, precision for rare classes is low, indicating that classical ML approaches may require augmentation with more sophisticated techniques for production deployment.

Chapter 5

Security, Implications, and Considerations

This chapter discusses the practical implications of our findings for the deployment of machine learning-based intrusion detection systems, along with security and ethical considerations.

5.1 Deployment Considerations

The experimental findings underscore a fundamental **"No Free Lunch"** theorem in NIDS methodology: optimizing for the recall of rare attacks (e.g., Shellcode) inherently incurs a cost in precision.

5.1.1 The Precision-Recall Trade-off

Our analysis of the *Shellcode* sink (94% Recall, 22% Precision) presents a distinct operational choice:

- **Type 1 Security (High Recall)**: Deploying XGB-S1 ensures that 19 out of 20 Shellcode attacks are caught. However, 80% of alerts will be false alarms (exploit noise, fuzzers). This fits environments where the cost of a breach is catastrophic (e.g., Critical Infrastructure).
- **Type 2 Security (High Precision)**: Deploying RF-S0 reduces false alarms but misses 50% of attacks. This is suitable only for environments with limited analyst capacity where alert fatigue must be minimized.

5.1.2 The Feature Engineering Imperative

The confounding of *Analysis* and *Backdoor* traffic (Confusion Ratio 2.37) demonstrates that current feature sets (NetFlow/Packet header derivatives) are insufficient to distinguish these attack topologies. Relying on "better models" (hyperparameter tuning) is futile here. Future deployments must incorporate:

- **Payload Inspection**: Deep Packet Inspection (DPI) features may resolve the ambiguity between Analysis probing and Backdoor channels.
- **Temporal Sequencing**: Analyzing the sequence of packets rather than individual flows could differentiate the persistent nature of a Backdoor from the bursty nature of Analysis scans.

5.1.3 Computational Costs

Training times varied significantly across our experiments. Oversampling (S2a) approximately doubled training time compared to class weighting (S1) due to the expanded training set. For real-time applications requiring frequent model updates, class weighting is preferable as it achieves comparable performance without additional data generation overhead.

5.2 Security Implications

5.2.1 Risks of False Negatives

Our baseline experiments (S0) demonstrate the critical danger of deploying models without imbalance handling: *zero detection* of rare attacks. In operational terms, this means:

- Worm propagation could proceed undetected, potentially compromising entire network segments.
- Shellcode injection attacks—often used as initial access vectors—would bypass detection entirely.
- Backdoor installations would establish persistent access for attackers without alerting defenders.

These rare attacks, while infrequent, often carry the highest impact. A single undetected backdoor can compromise an entire organization [2].

5.2.2 Cost of False Positives

While improved recall is desirable, the low precision observed for rare classes (3–22%) implies elevated false positive rates. In practice, this creates:

- **Alert Fatigue**: Security analysts may become desensitized to frequent false alarms, potentially missing genuine threats.
- **Operational Overhead**: Each alert requires investigation, consuming analyst time and organizational resources.

Balancing these trade-offs requires careful calibration to the organization's risk tolerance and available resources.

5.3 Ethical Considerations

5.3.1 Dual-Use Concerns

The techniques and findings presented in this research have potential dual-use implications. Understanding how IDS models fail to detect certain attack categories could, in principle, inform adversarial strategies for evading detection. We have mitigated this risk by:

- Focusing on well-documented public datasets rather than proprietary threat intelligence.
- Presenting findings in terms of model improvement rather than evasion techniques.
- Publishing reproducible methodology to enable defensive research.

5.3.2 Privacy in Network Monitoring

Intrusion detection inherently involves monitoring network traffic, which may contain sensitive information. Organizations deploying IDS solutions must:

- Ensure compliance with applicable privacy regulations (e.g., GDPR, CCPA).
- Implement appropriate data retention and access control policies.
- Consider privacy-preserving techniques such as differential privacy or federated learning for sensitive environments [21].

5.3.3 Bias in Detection

Our analysis reveals that models trained without imbalance handling systematically ignore minority classes. This represents a form of algorithmic bias: the model "learns" that certain attack types do not exist, simply because they are rare in the training data. This finding underscores the importance of explicitly addressing class imbalance to ensure equitable detection across all threat categories.

Chapter 6

Conclusion

This chapter summarizes our contributions, acknowledges limitations, and outlines directions for future research.

6.1 Summary

This study investigated the impact of class imbalance on machine learning-based intrusion detection using the UNSW-NB15 dataset. We systematically compared three classical models—Logistic Regression, Random Forest, and XGBoost—across three imbalance-handling strategies for both binary and multiclass classification tasks. Our 18-experiment evaluation yielded several significant findings:

Addressing Research Questions:

- **RQ1 (Imbalance Impact)**: Class imbalance severely degrades multiclass detection, with baseline models achieving 0% recall on all four rare attack categories despite high overall accuracy.
- **RQ2 (Strategy Effectiveness)**: Both class weighting and oversampling dramatically improve minority class detection. Class weighting increased Worms recall from 0% to 84% and Shellcode recall from 0% to 94%.
- **RQ3 (Model Consistency)**: All three models benefited from imbalance handling, though XGBoost consistently outperformed alternatives across both tasks and all strategies.
- **RQ4 (Rare Class Trade-offs)**: Improved recall for rare classes comes at the cost of reduced precision, representing an inherent precision-recall trade-off in imbalanced classification.

Key Contributions:

1. We provided the first systematic comparison of three imbalance strategies across three classical ML models on both binary and multiclass UNSW-NB15 tasks (18 experiments).

2. We demonstrated that aggregate accuracy metrics can be misleading, advocating for G-Mean and per-class metrics in IDS evaluation.
3. We established a reproducible baseline pipeline with fixed hyperparameters and documented methodology for future research comparison.
4. We quantified the critical failure of baseline models on rare attacks, providing empirical evidence for the necessity of imbalance handling in security-critical applications.

6.2 Limitations

We acknowledge several limitations of this study:

- **Single Dataset**: All experiments were conducted on UNSW-NB15. While this is a standard benchmark, results may not generalize to other network environments or datasets (e.g., CIC-IDS2017, BoT-IoT).
- **Fixed Hyperparameters**: We did not perform hyperparameter optimization, instead using fixed configurations to ensure fair comparison. Tuned models may achieve higher performance.
- **Computational Constraints**: While Binary and Logistic Regression experiments used 5 random seeds to ensure stability, Multiclass Tree-based models were limited to fewer seeds due to resource constraints. Results may exhibit minor variance.
- **Classical ML Only**: We did not compare against deep learning approaches (CNN, LSTM, Transformers), which may offer superior performance on complex network traffic patterns [12].
- **Low Rare Class Precision**: Despite improved recall, precision for rare classes remains low (3–22%), resulting in elevated false positive rates that may limit practical deployment.
- **No Temporal Analysis**: Network traffic is inherently time-series data, but our models treat each connection independently without considering temporal patterns.

6.3 Future Work

Several promising directions emerge from this research:

- **Hyperparameter Optimization**: Applying Bayesian optimization or automated machine learning (AutoML) to identify optimal model configurations.
- **Advanced Sampling Strategies**: Evaluating hybrid techniques such as SMOTE-ENN or SMOTE-Tomek that combine oversampling with noise reduction [17].

- **Deep Learning Comparison**: Benchmarking classical models against modern architectures including Convolutional Neural Networks, Recurrent Neural Networks, and Transformer-based models [13].
- **Multi-Dataset Validation**: Replicating experiments on CIC-IDS2017, CSE-CIC-IDS2018, and BoT-IoT datasets to assess generalizability [18].
- **Cost-Sensitive Deep Learning**: Exploring focal loss and other cost-sensitive objectives for neural network-based IDS.
- **Real-Time Deployment**: Evaluating inference latency and developing optimized models suitable for production network monitoring.
- **Adversarial Robustness**: Testing model resilience against adversarial examples designed to evade detection.

In conclusion, this study demonstrates that addressing class imbalance is not optional but essential for reliable intrusion detection. Our reproducible baseline and comprehensive evaluation provide a foundation for future research advancing the state of the art in machine learning-based network security.

References

[1] Nour Moustafa and Jill Slay. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *2015 military communications and information systems conference (MilCIS)*, pages 1–6. IEEE, 2015.

[2] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. A survey of network-based intrusion detection data sets. *Computers & security*, 86:147–167, 2019.

[3] Nour Moustafa and Jill Slay. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 2016.

[4] Vaishnavi Shanmugam, Roozbeh Razavi-Far, and Ehsan Hallaji. Addressing class imbalance in intrusion detection: a comprehensive evaluation of machine learning approaches. *Electronics*, 14(1):69, 2024.

[5] Sikha Bagui and Kunqi Li. Resampling imbalanced data for network intrusion detection datasets. *Journal of Big Data*, 8(1):6, 2021.

[6] Razan Abdulhammed, Hassan Musafer, Ali Alessa, Miad Faezipour, and Abdelshakour Abuzneid. Features dimensionality reduction approaches for machine learning based network intrusion detection. *Electronics*, 8(3), 2019.

[7] Nguyen Thai-Nghe, Zeno Gantner, and Lars Schmidt-Thieme. Cost-sensitive learning methods for imbalanced data. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2010.

[8] Rifkie Primartha and Bayu Adhi Tama. Anomaly detection using random forest: A performance revisited. In *2017 International conference on data and software engineering (ICoDSE)*, pages 1–6. IEEE, 2017.

[9] Uzma Amin, Aamir S Ahanger, F Masoodi, and AM Bamhdi. Ensemble based effective intrusion detection system for cloud environment over unsw-nb15 dataset. In *Scrs Conf. Proc. Intell. Syst*, pages 483–494, 2021.

[10] Shweta More, Moad Idrissi, Haitham Mahmoud, and A Taufiq Asyhari. Enhanced intrusion detection systems performance with unsw-nb15 data analysis. *Algorithms*, 17(2):64, 2024.

[11] Sydney M Kasongo and Yanxia Sun. Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset. *Journal of Big Data*, 7(1):105, 2020.

[12] R. Vinayakumar, Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7:41525–41550, 2019. Received December 27, 2018, accepted January 3, 2019, date of current version April 11, 2019.

[13] Amol D. Vibhute. Network anomaly detection and performance evaluation of convolutional neural networks on unsw-nb15 dataset. *Procedia Computer Science*, 235:261–268, 2024.

[14] Mithilesh Kumar Choudhary and Atul Kumar Mishra. Review paper on imbalanced network based intrusion detection system using deep learning technique. *International Journal of Advanced Research and Multidisciplinary Trends (IJARMT)*, 2(4):257–264, 2025.

[15] Tharmini Janarthanan and Shahrzad Zargari. Feature selection in unsw-nb15 and kddcup'99 datasets. In *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, pages 1881–1886, 2017.

[16] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. A Survey of Network-based Intrusion Detection Data Sets. *arXiv e-prints*, page arXiv:1903.02460, March 2019.

[17] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[18] Gozde Karatas, Onder Demir, and Ozgur Koray Sahingoz. Increasing the performance of machine learning-based idss on an imbalanced and up-to-date dataset. *IEEE Access*, 8:32150–32162, 2020.

[19] Elham Abdullah Al-Qarni. Addressing imbalanced data in network intrusion detection: A review and survey. *International Journal of Advanced Computer Science and Applications*, 15(2), 2024.

[20] Syed Ibrahim Imtiaz, Liaqat Ali Khan, Ahmad S. Almadhor, Sidra Abbas, Shtwai Alsubai, Michal Gregus, and Zunera Jalil. Efficient approach for anomaly detection in internet of things traffic using deep learning. *Concurrency and Computation: Practice and Experience*, 34(26):e8266347, 2022.

[21] Ghada Abdelmoumin, Jessica Whitaker, Danda B Rawat, and Abdul Rahman. A survey on data-driven learning for intelligent network intrusion detection systems. *Electronics*, 11(2):213, 2022.

[22] Paulina Arregoces, Jaime Vergara, Sergio Armando Gutiérrez, and Juan Felipe Botero. Network-based intrusion detection: A one-class classification approach. In *NOMS*, volume 2022, pages 1–6, 2022.