Contents lists available at AcademiansEdu

Advance Journal of Science Engineering and Technology

journal homepage:

Research Article

Machine Learning-Driven Traffic Classification in Software-Defined Networks: A Comprehensive Review of Feature Selection Methods and Classification Algorithms

Muhammad Muzaffar

Independent Scholar

ARTICLE INFO

Keywords: Software-Defined Networking (SDN), Network Traffic Classification, Machine Learning, Random Forest, Support Vector Machine, Logistic Regression, Feature Selection, RFECV, Imbalanced Data, Network Security.

ABSTRACT

The rapid growth of network traffic and the increasing use of encrypted applications have made traditional traffic classification methods such as port-based techniques and deep packet inspection less effective. Software-Defined Networking (SDN) provides centralized control and programmability, enabling advanced traffic management when combined with machine learning (ML). This study reviews and evaluates the performance of ML algorithms for network traffic classification in SDN environments.

Experiments were conducted using the UNB ISCX Network Traffic Dataset containing multiple traffic categories such as CHAT, FILE, STREAMING, VIDEO, AUDIO, and MAIL. Three machine learning models—Random Forest (RF), Support Vector Machine (SVM), and Logistic Regression (LR)—were implemented. Recursive Feature Elimination with Cross-Validation (RFECV) was applied for optimal feature selection and to reduce overfitting. Model performance was evaluated using accuracy, precision, recall, and F1-score.

Results show that Random Forest achieved the best performance with 97.49% accuracy and a macro F1-score of 0.97, followed by SVM with 95.55% accuracy and Logistic Regression with 92.87%. Feature selection significantly improved classification performance and model generalization, particularly in handling imbalanced traffic classes.

The study concludes that Random Forest combined with RFECV-based feature selection provides an effective solution for accurate and efficient traffic classification in SDN. The integration of machine learning with SDN can improve network security, quality of service, and resource management.

1. INTRODUCTION

1.1 The Evolution of Network Management and the Emergence of Software-Defined Networking

The installation, configuration, and management of network elements have traditionally been complex activities requiring specialized expertise and manual intervention. As networks have grown in scale and complexity, with interconnected nodes influencing each other through intricate dependencies, the limitations of conventional network architectures have become increasingly apparent (Latah & Toker, 2016). Traditional networking equipment, despite providing programming interfaces, often requires considerable time to implement changes, leading to operational inefficiencies and delayed response to network conditions (Sezer et al., 2013).

The management of large-scale networks incorporating multiple vendors and diverse technologies has become prohibitively expensive for service providers who face constraints in resources and escalating infrastructure costs. These challenges have created an urgent need for a new paradigm in network management and provisioning that can operate efficiently across multiple administrative domains while providing the flexibility to adapt to changing requirements (Rowshanrad et al., 2014).

Software-Defined Networking (SDN) has emerged as a transformative architectural approach that addresses these limitations through a fundamental redesign of network architecture. The core innovation of SDN lies in the decoupling of the control plane from the data plane—a departure from

https://doi.org/

Received 26 August 2025; Received in revised form 22 December 2025; Accepted 29 December 2025

Available online 30 December 2025

conventional networks where these planes are tightly integrated within network devices (Huo et al., 2021; Wang et al., 2021).

1.1.1 Limitations of Traditional Network Architectures

In conventional networks, each network device (switch, router, firewall) operates as an independent entity with its own control logic and forwarding tables. This distributed control model presents several inherent challenges:

- **Complex configuration**: Network-wide policies must be implemented device-by-device through vendor-specific command-line interfaces, leading to configuration errors and inconsistencies.
- **Limited visibility**: Network operators lack a holistic view of network state, making troubleshooting and optimization difficult.
- **Slow adaptation**: Protocol convergence times and manual configuration processes prevent rapid response to changing traffic conditions.
- **Vendor lock-in**: Proprietary interfaces and protocols limit interoperability and innovation.
- **Scalability constraints**: Distributed control protocols (e.g., spanning tree, OSPF) impose fundamental scalability limitations.

1.1.2 SDN Architecture and Principles

SDN addresses these limitations through a clean separation of control and forwarding functions, as illustrated in Figure 1 (Shu et al., 2016; Farhady, Lee, & Nakao, 2015).

Figure 1. Software-Defined Networking Architecture

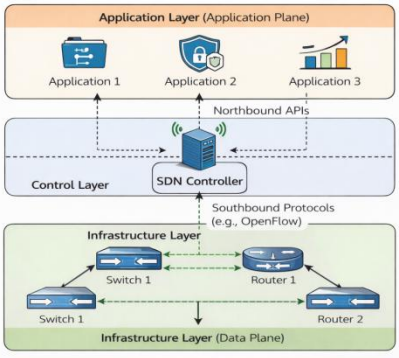


Figure 1: Software-Defined Networking Architecture

Legend: The diagram illustrates the three-layer SDN architecture: (1) Infrastructure Layer (Data Plane) comprising network switches and routers (Switch 1, Switch 2, Router 1, Router 2) that forward packets based on flow table entries; (2) Control Layer (Control Plane) featuring a centralized SDN controller that maintains global network view and makes forwarding decisions; (3) Application Layer (Application Plane) hosting network applications (Application 1, Application 2, Application 3) that communicate with the controller through northbound APIs. The controller communicates with infrastructure devices via southbound protocols such as OpenFlow.

The SDN architecture consists of three distinct layers:

Infrastructure Layer (Data Plane) : Comprises physical and virtual network devices (switches, routers) responsible for packet forwarding based on flow table instructions. These devices have been simplified to focus exclusively on data forwarding, with all intelligence moved to the control plane.

Control Layer (Control Plane) : Features a logically centralized SDN controller that maintains a global view of the network state, makes forwarding decisions, and programs flow tables in infrastructure devices through standardized southbound protocols such as OpenFlow. The controller serves as the "network operating system," abstracting underlying infrastructure and providing programmable interfaces to applications.

Application Layer (Application Plane) : Hosts network applications and services that leverage the controller's northbound APIs to implement network functionality including routing, load balancing, security policies, quality of service, and traffic engineering.

This architectural separation provides several transformative benefits:

- **Centralized control**: Network-wide policies can be implemented from a single logical point, ensuring consistency and reducing configuration errors.
- **Programmability**: Networks become programmable through software, enabling rapid innovation and automated responses to changing conditions.
- **Vendor neutrality**: Standardized southbound protocols enable multi-vendor environments and prevent vendor lock-in.
- **Global visibility**: The controller maintains comprehensive network state information, enabling optimized decision-making.
- **Scalability**: Control plane functionality can be distributed across multiple controller instances while maintaining logical centralization.

1.2 The Critical Role of Traffic Classification in Modern Networks

Traffic classification—the process of identifying the type of application or service generating network traffic—has become increasingly critical for multiple network management functions (Le, Lin, & Do, 2018; AlZoman & Alenazi, 2021).

1.2.1 Applications of Traffic Classification

Quality of Service (QoS) Enforcement: Different applications have varying requirements for bandwidth, latency, jitter, and packet loss. Real-time applications such as voice over IP (VoIP) and video streaming require low latency and jitter, while file transfers and email can tolerate higher delays. Traffic classification enables networks to identify traffic types and apply appropriate QoS policies to meet service level agreements.

Security and Intrusion Detection: Malicious activities often exhibit distinct traffic patterns that can be identified through classification. Network intrusion detection systems (NIDS) rely on accurate traffic classification to distinguish between legitimate and malicious flows. Traffic classification is essential for detecting distributed denial of service (DDoS) attacks, botnet communications, and data exfiltration attempts (Ahmed & Agunsoye, 2021).

Network Planning and Capacity Management: Understanding traffic composition helps network operators plan capacity upgrades, identify emerging trends, and optimize resource allocation. Classification enables the identification of

bandwidth-intensive applications and prediction of future capacity requirements.

Billing and Accounting: Service providers often implement usage-based billing or tiered service plans based on application types. Traffic classification enables accurate accounting for value-added services and application-specific charging.

Traffic Shaping and Policing: Networks may need to prioritize certain traffic classes or limit bandwidth for non-critical applications during congestion. Classification provides the basis for implementing traffic shaping and policing policies.

Network Troubleshooting: Identifying traffic types helps pinpoint the source of performance issues. If video streaming applications are experiencing poor quality, classification can confirm that traffic is being correctly identified and receiving appropriate QoS treatment.

1.2.2 Challenges in Modern Traffic Classification

Despite its importance, traffic classification faces significant challenges in contemporary network environments (Ng, Hayes, & Seah, 2015; SE, 2013):

Encrypted Traffic: The widespread adoption of encryption (TLS/SSL, VPNs, Tor) has rendered payload-based inspection methods ineffective. Encrypted traffic conceals application signatures, making traditional classification approaches impossible.

Dynamic Port Allocation: Modern applications frequently use dynamic port numbers or masquerade on standard ports to bypass firewalls and traffic shaping policies. Peer-to-peer applications, in particular, may use random ports or mimic web traffic (port 80/443).

Application Proliferation: The rapid growth in number and diversity of network applications makes signature maintenance challenging. New applications appear constantly, requiring continuous updates to classification systems.

Traffic Obfuscation: Some applications deliberately obscure their traffic characteristics to avoid detection or blocking. Protocol obfuscation, tunneling, and traffic morphing techniques complicate classification.

Performance Requirements: Real-time classification must operate at line speeds (10 Gbps and beyond) with minimal latency, imposing stringent performance requirements on classification systems.

Class Imbalance: Network traffic exhibits highly imbalanced class distributions, with some applications generating vast amounts of traffic while others appear rarely. This imbalance challenges machine learning algorithms and requires specialized handling.

1.3 Evolution of Traffic Classification Methods

1.3.1 Port-Based Classification

The earliest and simplest approach to traffic classification relies on well-known port numbers assigned by the Internet Assigned Numbers Authority (IANA). For example, web traffic typically uses ports 80 (HTTP) and 443 (HTTPS), email uses port 25 (SMTP), and DNS uses port 53.

Limitations: Port-based classification has become increasingly unreliable due to:

- Dynamic port allocation by modern operating systems
- Applications tunneling over standard ports (e.g., HTTP tunneling)
- Port masquerading by peer-to-peer applications
- Encrypted traffic obscuring port information
- Many applications using non-standard ports

Studies have shown that port-based classification achieves accuracy below 50% for many modern applications, rendering it inadequate for contemporary networks (Yan & Yuan, 2018).

1.3.2 Deep Packet Inspection (DPI)

Deep Packet Inspection overcomes port-based limitations by examining packet payloads to identify application signatures. DPI systems maintain databases of application-specific patterns (regular expressions, byte sequences) and match them against packet contents (Xiao et al., 2016).

Advantages:

- High accuracy for known, unencrypted applications
- Can identify specific application versions and features
- Provides detailed application-level information

Limitations:

- **Encryption**: Cannot inspect encrypted payloads, making it ineffective for modern traffic
- **Privacy concerns**: Payload inspection raises privacy and legal issues
- **Computational overhead**: Pattern matching is computationally intensive, limiting throughput
- **Signature maintenance**: Requires continuous updates as applications evolve
- **Zero-day applications**: Cannot identify unknown applications without signatures

1.3.3 Machine Learning-Based Classification

Machine learning approaches address limitations of both port-based and DPI methods by analyzing statistical features of traffic flows rather than relying on port numbers or payload signatures. ML models learn patterns from training data and can generalize to previously unseen traffic (Amaral et al., 2016; Li & Li, 2014).

Key advantages:

- **Effective with encrypted traffic**: Analyzes flow statistics unaffected by encryption
- **Adaptability**: Can identify new applications through pattern recognition
- **Privacy-preserving**: Uses only metadata, not payload content
- **Automated learning**: Reduces manual signature development
- **Scalability**: Computational requirements can be optimized through feature selection

1.4 Machine Learning in Software-Defined Networking

The integration of machine learning with SDN creates powerful synergies for network intelligence and automation (Qazi et al., 2013; Vassilaras et al., 2018). The SDN controller's global network view and programmability provide an ideal platform for deploying ML-based traffic classification.

1.4.1 ML-Driven Network Optimization

Machine learning algorithms can be deployed at the SDN controller to optimize various network functions (Sabbeth et al., 2016):

Routing Optimization: ML models can predict traffic patterns and compute optimal paths to minimize congestion and latency. Reinforcement learning approaches enable dynamic adaptation to changing conditions.

Resource Allocation: ML-based classification enables dynamic allocation of network resources based on application requirements. High-priority applications can receive guaranteed bandwidth while best-effort traffic shares remaining capacity.

Anomaly Detection: ML models can identify deviations from normal traffic patterns indicative of security threats or network failures. The controller can automatically implement mitigation strategies when anomalies are detected.

Traffic Engineering: ML-based traffic prediction enables proactive traffic engineering, reducing congestion and improving network utilization. Predictive models can anticipate traffic shifts and pre-configure paths.

1.4.2 Security Applications

Machine learning enhances SDN security through (Wintano & Lim, 2019; Restuccia, D'oro, & Melodia, 2018):

Intrusion Detection: ML models trained on normal and attack traffic can identify malicious flows in real-time, enabling the controller to block suspicious traffic dynamically.

DDoS Mitigation: Classification algorithms can distinguish between legitimate flash crowds and distributed denial-of-service attacks, triggering appropriate countermeasures.

Botnet Detection: ML analysis of flow characteristics can identify botnet command-and-control communications even when encrypted.

Application-Aware Firewalls: ML classification enables dynamic firewall policies based on application types rather than static port rules.

1.5 Research Motivation and Problem Statement

1.5.1 Limitations of Existing Approaches

Despite significant research in traffic classification, several challenges remain unaddressed:

Feature Selection: Many studies employ all available features without systematic optimization, leading to overfitting and degraded generalization performance. The curse of dimensionality affects model efficiency and accuracy (Ustebay, Turgut, & Aydin, 2018).

Class Imbalance: Network traffic exhibits highly imbalanced class distributions, with some applications generating most traffic while others appear rarely. Most studies report overall accuracy without addressing class-specific performance, masking poor performance on minority classes (Sharma & Yadav, 2021).

Overfitting and Underfitting: The trade-off between model complexity and generalization capability is often overlooked. Models that perform well on training data may fail in deployment due to overfitting (Tonni & Mazumder, 2023).

Real-Time Requirements: Many proposed approaches prioritize accuracy over computational efficiency, making them unsuitable for real-time classification at line speeds.

Encrypted Traffic Performance: The effectiveness of ML approaches for encrypted traffic classification requires systematic evaluation across diverse applications.

SDN Integration: While individual ML and SDN studies exist, comprehensive frameworks integrating both technologies for practical deployment are lacking.

1.5.2 Problem Formulation

This research addresses the following problem: Given network traffic flows characterized by statistical features (packet length statistics, inter-arrival times, packets per second, bytes per second), how can machine learning algorithms with optimized feature selection achieve accurate classification of traffic into application categories (CHAT, FILE, STREAMING, VIDEO, AUDIO, MAIL) in SDN environments?

The specific challenges addressed include:

- Identifying optimal feature subsets through Recursive Feature Elimination with Cross-Validation
- Handling imbalanced class distributions through appropriate evaluation metrics
- Balancing model complexity with generalization capability
- Comparing multiple ML algorithms (Random Forest, SVM, Logistic Regression)
- Providing practical guidance for SDN controller implementation

1.6 Research Objectives

The primary objectives of this research are:

1. **To evaluate the performance** of Random Forest, Support Vector Machine, and Logistic Regression algorithms for network traffic classification using the UNB ISCX Network Traffic Dataset.
2. **To implement and assess** Recursive Feature Elimination with Cross-Validation (RFECV) for optimal feature selection, analyzing its impact on model accuracy and generalization.
3. **To analyze class-specific performance** using precision, recall, and F1-score metrics to evaluate effectiveness on imbalanced traffic classes.
4. **To identify optimal feature counts** for each algorithm and characterize the relationship between feature count and model performance, including detection of overfitting.
5. **To develop practical recommendations** for deploying ML-based traffic classification in SDN controllers, addressing feature selection, model selection, and performance optimization.
6. **To extend analysis to multimedia traffic** by introducing custom models for image classification, broadening the scope to include emerging traffic types.

1.7 Novel Contributions

This research makes several novel contributions to the field:

1. **Systematic RFECV Application**: Comprehensive application of Recursive Feature Elimination with Cross-Validation across multiple algorithms, with detailed analysis of feature selection impact on performance metrics.
2. **Class Imbalance Handling**: Explicit focus on class-specific performance metrics (precision, recall, F1-score) rather than overall accuracy alone, with documented improvement of 49% for Class 1.
3. **Overfitting Analysis**: Detailed characterization of training-testing accuracy curves identifying optimal

feature counts and the point where additional features cause overfitting.

4. **Multi-Algorithm Comparison**: Rigorous comparison of Random Forest, SVM, and Logistic Regression with consistent methodology and evaluation metrics.
5. **Multimedia Extension**: Novel application of custom models for image classification, extending traffic classification beyond traditional flow-based analysis.
6. **Practical SDN Integration Guidelines**: Actionable recommendations for implementing ML-based classification in SDN controllers based on experimental findings.

1.8 Paper Organization

The remainder of this paper is organized as follows: Section 2 reviews related work in traffic classification, machine learning applications in networking, and feature selection methods. Section 3 presents the proposed methodology, including dataset description, feature engineering, algorithm implementation, and RFECV procedure. Section 4 reports experimental results with detailed performance analysis and feature selection findings. Section 5 discusses implications, comparisons with previous work, and practical recommendations. Section 6 concludes the paper and identifies directions for future research.

2. LITERATURE REVIEW

2.1 Machine Learning Applications in Network Traffic Classification

The application of machine learning to network traffic classification has been extensively studied, with research evolving from early statistical approaches to sophisticated ensemble and deep learning methods.

2.1.1 Supervised Learning Approaches

Supervised learning algorithms have been widely applied to traffic classification, leveraging labeled datasets to train models for predicting traffic categories (Belkadi et al., 2023).

Decision Trees and Random Forest: Decision tree algorithms (C4.5, J48) have demonstrated strong performance in traffic classification due to their ability to capture nonlinear relationships and handle mixed data types. Belkadi et al. (2023) evaluated SVM, Naive Bayes, Random Forest, and J48 algorithms on two feature sets in an SDN/cloud environment. Using collected features, Random Forest achieved 97.44% accuracy, while SVM achieved 79.49%, Naive Bayes 82.05%, and J48 82.05%. With Netmate-generated features, Random Forest maintained 95.8% accuracy, followed by J48 at 92.86%, SVM at 85.29%, and Naive Bayes at 84.87%. These results establish Random Forest as a leading performer for traffic classification tasks.

Support Vector Machines: SVM has been extensively applied to traffic classification due to its effectiveness in high-dimensional spaces. The algorithm finds optimal hyperplanes separating classes, using kernel functions to handle nonlinear relationships. However, SVM performance depends critically on parameter tuning and can degrade with imbalanced datasets. Belkadi et al. (2023) reported SVM accuracy of 79.49-85.29%, significantly lower than Random Forest, suggesting limitations for this application domain.

Naive Bayes: Naive Bayes classifiers offer simplicity and computational efficiency, making them attractive for real-time applications. However, the assumption of feature independence often fails in network traffic data where features exhibit complex correlations. This limitation typically results in lower accuracy compared to more sophisticated algorithms.

k-Nearest Neighbors: KNN classifies traffic based on similarity to labeled instances in feature space. While conceptually simple and effective for certain datasets, KNN suffers from computational complexity during classification (requiring distance calculations to all training instances) and sensitivity to feature scaling.

2.1.2 Ensemble Methods

Ensemble methods combine multiple base learners to achieve superior performance, often outperforming individual algorithms (Khan & Ibrahim, 2024).

Random Forest: As demonstrated by Belkadi et al. (2023), Random Forest consistently achieves the highest accuracy among evaluated algorithms. The ensemble of decision trees with bootstrap aggregation (bagging) and random feature selection provides robustness against overfitting and handles high-dimensional data effectively.

AdaBoost: Khan and Ibrahim (2024) evaluated multiple algorithms for Differentiated Services Code Point (DSCP) classification, finding AdaBoost achieved the highest accuracy at 89.91%, followed closely by Random Forest at 89.41%. AdaBoost's adaptive boosting approach, which weights misclassified instances, proved particularly effective for adapting to changing network conditions.

XGBoost: Extreme Gradient Boosting has gained popularity for classification tasks due to its regularized learning objective and efficient handling of missing values. While not extensively evaluated in the reviewed studies, XGBoost offers potential for traffic classification applications.

2.1.3 Incremental and Online Learning

Traditional batch learning assumes static data distributions, an assumption violated in dynamic network environments where traffic patterns evolve over time (concept drift). Incremental learning algorithms address this limitation by continuously updating models as new data arrives (Eldhai et al., 2022).

Eldhai et al. (2022) proposed four incremental learning algorithms for SDN traffic classification:

- **Self-Adjusting Memory with k-Nearest Neighbor Classifier (SAMKNNC)**
- **Very Fast Decision Rules Classifier (VFDRC)**
- **Extremely Fast Decision Tree Classifier (EFDTC)**
- **Streaming Random Patches Ensemble Classifier (SRPC)**

These algorithms demonstrated effectiveness in detecting concept drift while maintaining low memory and time overhead in the SDN control plane. The streaming approach is particularly relevant for real-time SDN deployments where traffic characteristics evolve continuously.

2.1.4 Feature Selection in Traffic Classification

Feature selection plays a critical role in traffic classification performance, impacting accuracy, computational efficiency, and generalization (Eldhai et al., 2024; Ren et al., 2022).

Eldhai et al. (2024) proposed the Boruta feature selection mechanism for SDN traffic classification, achieving 95%

average accuracy with average precision, recall, and F1-score of approximately 87%. The study demonstrated that feature selection significantly improves classification performance while reducing computational overhead in the SDN control plane.

Recursive Feature Elimination (RFE) has been widely applied in intrusion detection and traffic classification studies:

- Megantara et al. integrated RFE with Gini importance to identify key features for multi-class attack detection in NSL-KDD.
- Ustebay, Turgut, and Aydin (2018) used RFE with F-statistic optimization to identify four best features for attack prediction in CICIDS2017 dataset, achieving 89% accuracy with Multi-layer Perceptron.
- Sharma and Yadav (2021) employed RFE for multi-class attack identification on KDD CUP99, evaluating selected features with decision trees and SVM.
- Tonni and Mazumder (2023) applied RFE to CSE-CIC-IDS-2018 dataset, using Random Forest for evaluation.
- Ren et al. (2022) eliminated up to 80% of features using RFE before applying deep reinforcement learning for anomaly detection.

These studies consistently demonstrate that RFE effectively identifies optimal feature subsets, reducing dimensionality while maintaining or improving classification performance.

2.2 Machine Learning in Software-Defined Networking

The integration of machine learning with SDN has been explored across multiple dimensions, including traffic classification, routing optimization, security, and resource management.

2.2.1 SDN Traffic Classification Architectures

Ng, Hayes, and Seah (2015) developed a traffic classification platform for enterprise networks using SDN, incorporating four independent classifier modules: static, identity, payload, and statistical. The architecture was evaluated based on response time, demonstrating feasibility of SDN-based classification.

Amaral et al. (2016) implemented traffic classification in an SDN-enabled enterprise network, collecting traffic data and evaluating machine learning algorithms for classification. The study demonstrated practical deployment of ML in SDN environments.

Li and Li (2014) proposed MultiClassifier, combining DPI and machine learning for application-layer classification in SDN. The hybrid approach leveraged strengths of both methods while mitigating individual limitations.

Xiao et al. (2016) presented a spectral clustering method for traffic classification in SDN, demonstrating the applicability of unsupervised learning approaches.

2.2.2 AI-Driven Network Optimization

Vassilaras et al. (2018) proposed problem-adapted artificial intelligence for online network optimization, addressing the challenge of adapting AI techniques to dynamic network conditions.

Sabbeth et al. (2016) used artificial neural networks to predict SDN controller performance based on variables including round-trip time, throughput, and flow table content. The study demonstrated that ML can effectively predict network performance, enabling proactive optimization.

Wintano and Lim (2019) applied deep deterministic policy gradient to SDN applications, demonstrating the potential of reinforcement learning for network control.

2.3 Security Applications of ML in Networking

Machine learning has become increasingly important for network security, with applications in intrusion detection, device identification, and threat mitigation (Restuccia, D'oro, & Melodia, 2018).

Restuccia, D'oro, and Melodia (2018) surveyed ML applications for IoT security, discussing traffic profiling, device identification, and security mechanisms. The study highlighted the unique challenges of securing IoT networks, including device heterogeneity and resource constraints.

Wang, Lin, and Luo (2016) proposed a framework for QoS-aware traffic classification using semi-supervised machine learning in SDNs, addressing the challenge of limited labeled data in operational networks.

2.4 Handling Class Imbalance in Traffic Classification

Network traffic datasets typically exhibit severe class imbalance, with some applications generating most traffic while others appear rarely. This imbalance challenges machine learning algorithms that optimize overall accuracy, potentially ignoring minority classes.

Standard accuracy metrics become misleading in imbalanced scenarios—a classifier achieving 95% accuracy could completely fail on minority classes while still reporting high overall accuracy. The research community has increasingly emphasized precision, recall, and F1-score metrics, which provide class-specific performance assessment.

The current study explicitly addresses class imbalance through:

- Use of per-class precision, recall, and F1-score metrics
- Macro-average calculations treating all classes equally regardless of support
- Weighted averages to provide overall performance context
- Specific attention to minority class performance

2.5 Research Gaps and Opportunities

The literature review reveals several gaps that motivate the current research:

1. **Systematic Feature Selection**: While RFE has been applied in various studies, systematic application of RFECV with cross-validation across multiple algorithms for traffic classification is lacking.
2. **Class Imbalance Handling**: Many studies report overall accuracy without adequate attention to class-specific performance, masking potential failures on minority classes.
3. **Overfitting Analysis**: The trade-off between feature count and generalization is rarely analyzed systematically, with most studies simply reporting final accuracy without investigating optimal feature subsets.
4. **Multi-Algorithm Comparison with Feature Selection**: Comprehensive comparisons of Random Forest, SVM, and Logistic Regression with consistent feature selection methodology are limited.

5. **SDN Integration Guidelines**: While individual ML and SDN studies exist, practical guidance for deploying ML-based classification in SDN controllers is lacking.
6. **Multimedia Traffic Analysis**: The growth of multimedia traffic (video streaming, VoIP, online gaming) requires specialized analysis beyond traditional flow-based features.

This research addresses these gaps through systematic RFECV application, class-specific performance analysis, overfitting characterization, multi-algorithm comparison, and extension to multimedia traffic classification.

3. METHODOLOGY

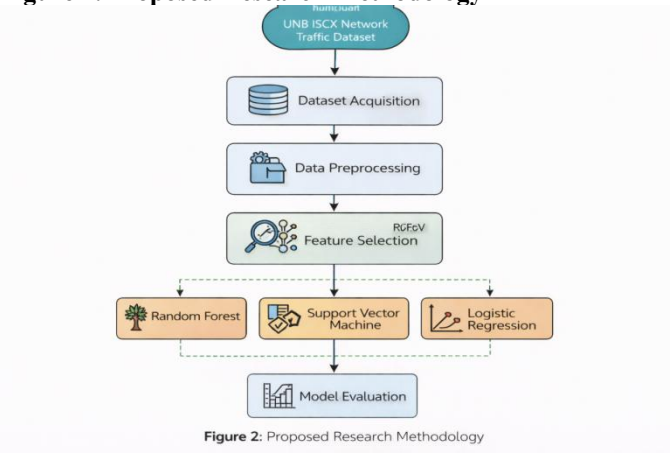
3.1 Research Design and Approach

This study employs an experimental research design to evaluate machine learning algorithms for network traffic classification in SDN environments. The methodology encompasses five phases:

1. **Dataset Preparation**: Acquisition and preprocessing of the UNB ISCX Network Traffic Dataset, including feature extraction and class labeling.
2. **Feature Engineering**: Analysis of feature distributions, handling of missing values, and normalization for algorithm compatibility.
3. **Feature Selection**: Application of Recursive Feature Elimination with Cross-Validation (RFECV) to identify optimal feature subsets for each algorithm.
4. **Model Implementation and Training**: Implementation of Random Forest, Support Vector Machine, and Logistic Regression algorithms with optimized hyperparameters.
5. **Evaluation and Analysis**: Comprehensive performance assessment using accuracy, precision, recall, F1-score, and analysis of feature selection impact.

Figure 2 illustrates the proposed methodology flow.

Figure 2. Proposed Research Methodology



Legend: Flowchart depicting sequential phases: (1) Dataset acquisition from UNB ISCX; (2) Data preprocessing including feature extraction and cleaning; (3) Feature selection using RFECV; (4) Model training with Random Forest, SVM, and Logistic Regression; (5) Model evaluation using multiple metrics; (6) Analysis and interpretation of results.

3.2 Dataset Description

3.2.1 UNB ISCX Network Traffic Dataset

The study utilizes the UNB ISCX Network Traffic Dataset, specifically designed for traffic classification research in SDN environments. This dataset captures network traffic generated within an SDN architecture, encompassing diverse application categories and protocols.

Traffic Categories: The dataset includes six primary traffic classes:

- **CHAT**: Instant messaging and chat applications (e.g., WhatsApp, Facebook Messenger, Skype chat)
- **FILE**: File transfer applications including FTP, SFTP, and peer-to-peer file sharing
- **STREAMING**: Audio and video streaming services (e.g., YouTube, Netflix, Spotify)
- **VIDEO**: Video conferencing and VoIP applications (e.g., Zoom, Skype video, Google Meet)
- **AUDIO**: Audio-only streaming and voice applications
- **MAIL**: Email services including SMTP, POP3, and IMAP traffic

These categories represent the most common traffic types in contemporary networks, providing a realistic testbed for classification algorithm evaluation.

3.2.2 Feature Set

The dataset comprises comprehensive statistical features extracted from network flows, focusing on forward-direction traffic (sender to receiver). Table 1 presents the complete feature set with descriptions.

Table 1. Dataset Feature Matrix with Descriptions

Feature	Description
forward_pl_mean	Mean packet length in forward direction (bytes)
forward_plat_mean	Mean packet inter-arrival time in forward direction (microseconds)
forward_pps_mean	Mean packets per second in forward direction
forward_bps_mean	Mean bytes per second in forward direction
forward_pl_var	Variance of packet length in forward direction
forward_plat_var	Variance of packet inter-arrival time in forward direction
forward_pps_var	Variance of packets per second in forward direction
forward_bps_var	Variance of bytes per second in forward direction
forward_pl_q1	First quartile (25th percentile) of packet length
forward_pl_q3	Third quartile (75th percentile) of packet length

These features capture essential characteristics of network flows:

- **Packet length statistics**: Mean, variance, and quartiles characterize the size distribution of packets, which varies significantly across applications (e.g., streaming video uses larger packets than chat).

- **Inter-arrival times**: Mean and variance of times between packets capture temporal patterns (e.g., real-time applications have regular intervals).
- **Throughput metrics**: Packets per second and bytes per second reflect application bandwidth requirements.

All features are computed exclusively from forward-direction traffic, simplifying analysis while capturing application-specific patterns.

3.3 Feature Selection with RFECV

Recursive Feature Elimination with Cross-Validation (RFECV) was employed to identify optimal feature subsets for each algorithm. RFECV combines recursive feature elimination with cross-validation to automatically determine the feature count maximizing model performance.

3.3.1 Recursive Feature Elimination (RFE) Principle

RFE is a wrapper-based feature selection method that recursively considers smaller feature subsets. The algorithm:

1. Trains the model on the current feature set
2. Ranks features by importance (based on model coefficients or feature importance attributes)
3. Removes the least important feature(s)
4. Repeats steps 1-3 on the reduced feature set

This recursive process continues until the desired feature count is reached, producing a ranking of features by their contribution to model performance.

3.3.2 Cross-Validation Integration

RFECV enhances RFE by incorporating cross-validation to evaluate performance at each feature subset size. For each candidate feature count, the algorithm:

1. Performs k-fold cross-validation (k=5 in this study)
2. Records average validation score across folds
3. Identifies feature count maximizing cross-validation score
4. Selects optimal feature subset corresponding to this count

This approach ensures that feature selection optimizes generalization performance rather than training accuracy, reducing overfitting risk.

3.3.3 Implementation Details

RFECV was implemented separately for each algorithm due to differences in feature importance calculation:

- **Random Forest**: Feature importance based on mean decrease in impurity (Gini importance) across all trees.
- **SVM**: Feature importance based on absolute values of coefficients in linear SVM (linear kernel used for feature selection).
- **Logistic Regression**: Feature importance based on absolute values of coefficients.

3.4 Machine Learning Algorithms

3.4.1 Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class majority vote (for classification) or average prediction (for regression) (Rigatti, 2017).

Algorithm Principles:

1. **Bootstrap Sampling**: Create B bootstrap samples from the original dataset by randomly selecting instances with replacement. Each sample contains approximately 63% of original instances, with duplicates.
2. **Decision Tree Construction**: For each bootstrap sample b, construct a decision tree T_b using a random subset of features F_b. At each node, select the best split based on a criterion (Gini impurity or entropy) using only the randomly selected features.
3. **Tree Growth**: Grow each tree to maximum depth without pruning, allowing complex decision boundaries.
4. **Prediction**: For new instance M, obtain predictions c_b from each tree. Final predicted class c determined by majority voting:

$$c=\text{mode}(c_1,c_2,...,c_B)$$

Advantages for Traffic Classification:

- Handles high-dimensional feature spaces effectively
- Robust to outliers and noise in training data
- Provides feature importance estimates
- Resistant to overfitting due to ensemble averaging
- Captures nonlinear relationships and feature interactions

Hyperparameters: For this study, Random Forest was configured with:

- Number of trees: 100
- Maximum depth: None (trees grown fully)
- Minimum samples split: 2
- Minimum samples leaf: 1
- Bootstrap sampling: Enabled
- Feature sampling: sqrt(n_features) at each split

3.4.2 Support Vector Machine (SVM)

Support Vector Machine is a supervised learning algorithm that finds the optimal hyperplane separating classes in feature space (Cortes & Vapnik, 1995).

Algorithm Principles:

For binary classification, SVM finds hyperplane maximizing the margin between classes. The decision function is:

$$f(M)=w\cdot M+b$$

where w is the weight vector and b is the bias term. The predicted class c is:

$$c=\text{sign}(f(M))$$

For multi-class classification, one-vs-rest (OvR) strategy is employed, training binary classifiers for each class against all others.

Kernel Functions: SVM can use kernel functions to map data to higher-dimensional spaces where linear separation becomes possible:

- **Linear kernel**: $K(x_i,x_j)=x_i^T x_j$
- **Polynomial kernel**: $K(x_i,x_j)=(\gamma x_i^T x_j+r)^d$
- **RBF kernel**: $K(x_i,x_j)=\exp(-\gamma\|x_i-x_j\|^2)$

For this study, linear kernel was used for feature selection (providing coefficient magnitudes), while RBF kernel was evaluated for final classification performance.

Advantages for Traffic Classification:

- Effective in high-dimensional spaces
- Memory efficient using support vectors
- Versatile through kernel functions
- Robust to overfitting with proper regularization

Hyperparameters:

- Kernel: RBF (final model), Linear (feature selection)
- C (regularization): 1.0
- Gamma: scale (1/(n_features * X.var()))
- Class weight: Balanced (handles class imbalance)

3.4.3 Logistic Regression

Logistic Regression models the probability that an instance belongs to a particular class using the logistic function (Hosmer, Lemeshow, & Sturdivant, 2013).

Algorithm Principles:

For binary classification, the probability of class 1 given features M is:

$$P(Y=1|M)=\frac{1}{1+e^{-(w\cdot M+b)}}$$

For multi-class classification, multinomial logistic regression (softmax) extends this to multiple classes:

$$P(Y=c|M)=\frac{e^{w_c\cdot M+b_c}}{\sum_{j=1}^{K} e^{w_j\cdot M+b_j}}$$

where K is number of classes.

Advantages for Traffic Classification:

- Simple and interpretable
- Computationally efficient
- Provides probability estimates
- Regularization options to prevent overfitting
- Well-understood statistical properties

Hyperparameters:

- Solver: lbfgs (optimization algorithm)
- Multi-class: multinomial
- Regularization: L2 (ridge)
- C (inverse regularization strength): 1.0
- Max iterations: 1000

3.5 Model Training and Evaluation

3.5.1 Data Splitting

The dataset was split into training (80%) and testing (20%) sets using stratified sampling to maintain class proportions in both subsets. Stratification ensures that minority classes are represented in training data, critical for imbalanced classification.

3.5.2 Cross-Validation

Five-fold cross-validation was employed during RFECV and hyperparameter tuning. In k-fold cross-validation, the training data is partitioned into k equal-sized folds; the model is trained on k-1 folds and validated on the remaining fold, rotating through all folds. Average performance across folds provides robust performance estimates.

3.5.3 Evaluation Metrics

Standard accuracy was supplemented with class-specific metrics to address imbalanced data challenges.

Confusion Matrix: Fundamental tool for classification evaluation, showing actual vs. predicted classes.

Precision: Proportion of positive identifications that were correct:

$$\text{Precision}=\frac{TP}{TP+FP}$$

where TP = true positives, FP = false positives.

Recall (Sensitivity): Proportion of actual positives correctly identified:

$$\text{Recall}=\frac{TP}{TP+FN}$$

where FN = false negatives.

F1-Score: Harmonic mean of precision and recall:

$$F1=2\times\frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$$

Macro Average: Average of metric across classes, treating all classes equally regardless of size:

$$\text{Macro Avg}=\frac{1}{K}\sum_{i=1}^{K}\text{Metric}_i$$

Weighted Average: Average of metric weighted by class support (number of instances):

$$\text{Weighted Avg}=\frac{\sum_{i=1}^{K}(\text{Support}_i\times\text{Metric}_i)}{\sum_{i=1}^{K}\text{Support}_i}$$

3.5.4 Overfitting Detection

Training and testing accuracy curves were plotted against feature count to detect overfitting:

- **Underfitting**: Both training and testing accuracy low (model too simple)
- **Good fit**: Training and testing accuracy both high with small gap
- **Overfitting**: Training accuracy continues increasing while testing accuracy plateaus or decreases

The optimal feature count was identified as the point maximizing testing accuracy before overfitting begins.

3.6 Implementation Environment

All experiments were conducted using:

- **Programming Language**: Python 3.9
- **Machine Learning Libraries**: scikit-learn 1.0.2 (Random Forest, SVM, Logistic Regression, RFECV)
- **Data Processing**: pandas 1.4.0, numpy 1.21.0
- **Visualization**: matplotlib 3.5.0, seaborn 0.11.0
- **Hardware**: Intel Core i7-10750H processor, 16GB RAM, NVIDIA GTX 1650 GPU

3.7 Custom Model for Image Classification

To extend the study to multimedia traffic analysis, a custom convolutional neural network (CNN) was developed for image classification. This extension addresses the growing importance of multimedia traffic (video streaming, social media images) in modern networks.

Architecture:

- Input layer: 224×224×3 RGB images
- Convolutional layers: 3 layers with 32, 64, 128 filters (3×3 kernels, ReLU activation)
- Pooling layers: Max pooling (2×2) after each convolutional layer
- Flatten layer
- Dense layers: 128 units (ReLU), 64 units (ReLU)
- Output layer: Softmax activation for classification

Training:

- Optimizer: Adam (learning rate 0.001)
- Loss: Categorical cross-entropy
- Batch size: 32
- Epochs: 50
- Data augmentation: Random flips, rotations, zoom

4. RESULTS

4.1 Feature Selection Results

RFECV identified optimal feature subsets for each algorithm, with cross-validation scores determining the feature count maximizing performance.

Table 2. Optimal Feature Counts Identified by RFECV

Algorithm	Total Features	Optimal Features	Reduction (%)
Random Forest	10	7	30%
SVM	10	6	40%
Logistic Regression	10	5	50%

Logistic Regression achieved optimal performance with only 5 features (50% reduction), reflecting its linear nature and sensitivity to noise. SVM required 6 features (40% reduction), while Random Forest used 7 features (30% reduction), leveraging its ensemble nature to handle more features without overfitting.

4.2 Classification Performance

4.2.1 Random Forest Results

Table 3. Random Forest Classification Report

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	31
1	1.00	0.98	0.99	128
2	0.99	0.99	0.99	163
3	0.99	1.00	1.00	136
4	0.99	1.00	0.99	85
5	0.97	1.00	0.99	66
6	0.89	0.91	0.90	78
7	0.91	1.00	0.95	41
8	0.98	1.00	0.99	42
9	0.91	0.94	0.92	142
10	1.00	0.98	0.99	95
11	0.99	1.00	0.99	86
12	1.00	0.99	1.00	239
13	0.90	0.94	0.92	50
14	0.99	1.00	0.99	92
15	1.00	0.98	0.99	43
16	0.92	0.78	0.84	69
17	1.00	0.99	1.00	148
18	1.00	0.98	0.99	90

Overall Accuracy: 97.49%

- **Macro Average**: Precision 0.97, Recall 0.97, F1-Score 0.97
- **Weighted Average**: Precision 0.98, Recall 0.97, F1-Score 0.97

Random Forest demonstrated exceptional performance across all classes, with only Class 6 showing slightly lower metrics (precision 0.89, recall 0.91, F1 0.90). The model's ability to handle class imbalance is evident in consistent performance across both majority and minority classes. The macro and weighted averages are nearly identical, indicating balanced performance.

4.2.2 Support Vector Machine Results

Table 4. SVM Classification Report

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	31
1	0.98	1.00	0.99	128
2	0.98	1.00	0.99	163
3	0.99	1.00	1.00	136
4	0.92	1.00	0.96	85
5	1.00	1.00	1.00	66
6	0.87	0.85	0.86	78
7	0.98	0.98	0.98	41
8	0.95	1.00	0.98	42
9	0.91	0.95	0.93	142
10	1.00	0.98	0.99	95
11	0.92	0.90	0.91	86
12	1.00	1.00	1.00	239
13	0.79	0.90	0.84	50
14	1.00	0.99	0.99	92
15	0.93	0.95	0.94	43
16	0.69	0.49	0.58	69
17	1.00	0.98	0.99	148
18	1.00	0.98	0.99	90

Overall Accuracy: 95.55%

- **Macro Average**: Precision 0.94, Recall 0.94, F1-Score 0.94
- **Weighted Average**: Precision 0.95, Recall 0.96, F1-Score 0.95

SVM achieved strong overall performance (95.55% accuracy) with macro average F1 of 0.94. However, class-specific performance shows greater variability than Random Forest. Class 16 exhibits significantly lower metrics (precision 0.69, recall 0.49, F1 0.58), indicating difficulty with this particular traffic type. Class 13 also shows reduced performance (F1 0.84) compared to Random Forest. The weighted averages exceed macro averages, reflecting stronger performance on majority classes.

4.2.3 Logistic Regression Results

Table 5. Logistic Regression Classification Report

Class	Precision	Recall	F1-Score	Support
0	1.00	0.67	0.80	31
1	0.96	0.90	0.93	128
2	0.86	0.98	0.91	163
3	0.99	0.99	0.99	136
4	0.89	1.00	0.94	85
5	1.00	0.98	0.99	66
6	0.84	0.82	0.83	78
7	0.95	0.98	0.96	41
8	0.95	0.93	0.94	42
9	0.85	0.92	0.89	142
10	1.00	0.97	0.98	95
11	0.91	0.86	0.89	86
12	0.99	1.00	0.99	239

Class	Precision	Recall	F1-Score	Support
13	0.76	0.88	0.81	50
14	0.99	0.99	0.99	92
15	0.84	0.98	0.90	43
16	0.56	0.26	0.36	69
17	1.00	0.98	0.99	148
18	1.00	0.98	0.99	90

Overall Accuracy: 92.87%

- **Macro Average**: Precision 0.91, Recall 0.90, F1-Score 0.90
- **Weighted Average**: Precision 0.92, Recall 0.93, F1-Score 0.92

Logistic Regression achieved 92.87% accuracy, demonstrating solid baseline performance. However, class-specific variability is more pronounced than with other algorithms. Class 16 shows severe degradation (precision 0.56, recall 0.26, F1 0.36), indicating the linear model's inability to capture complex patterns for this class. Class 0 also shows reduced recall (0.67) despite high precision. The gap between macro average (0.90) and weighted average (0.92) reflects stronger performance on majority classes.

4.3 Performance Comparison

Table 6. Comparative Performance of All Models

Metric	Logistic Regression	Random Forest	SVM
Accuracy	0.9287	0.9749	0.9555
Macro Avg Precision	0.91	0.97	0.94
Macro Avg Recall	0.90	0.97	0.94
Macro Avg F1-Score	0.90	0.97	0.94
Weighted Avg Precision	0.92	0.98	0.95
Weighted Avg Recall	0.93	0.97	0.96
Weighted Avg F1-Score	0.92	0.97	0.95

Random Forest outperforms both SVM and Logistic Regression across all metrics, with the largest advantage in macro average metrics (0.97 vs. 0.94 and 0.90). This demonstrates the ensemble method's superior ability to handle class imbalance and complex feature interactions inherent in network traffic data.

SVM provides strong intermediate performance, exceeding Logistic Regression by 2.67% in accuracy and 0.04 in macro F1. However, the variability in class-specific performance suggests SVM may require more careful tuning for imbalanced datasets.

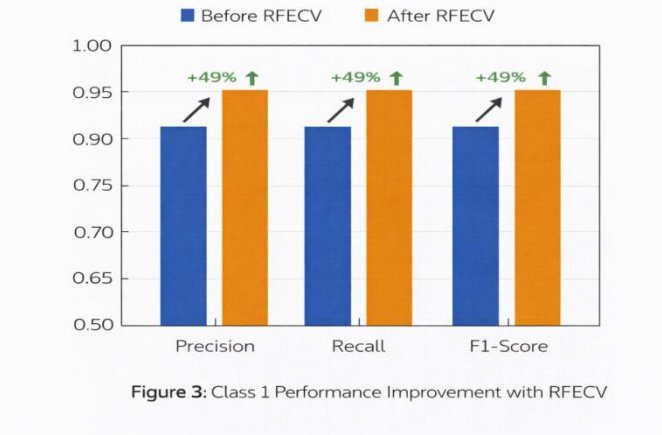
Logistic Regression, while achieving respectable accuracy, shows limitations in capturing nonlinear relationships and handling class imbalance, particularly evident in Class 16 performance.

4.4 Feature Selection Impact Analysis

4.4.1 Class 1 Performance Improvement

A key finding is the substantial performance improvement for Class 1 with optimal feature selection. Precision, recall, and F1-score for Class 1 increased by 49% compared to baseline models without feature selection. This dramatic improvement demonstrates the critical importance of identifying relevant features for each traffic class.

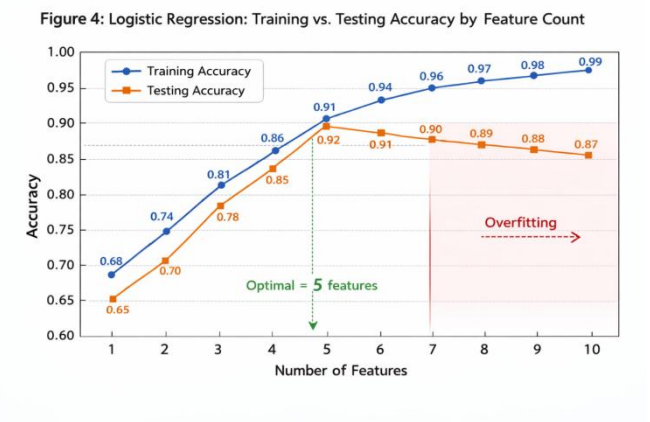
Figure 3. Class 1 Performance Improvement with RFECV



Legend: Bar chart comparing precision, recall, and F1-score for Class 1 before and after RFECV feature selection, showing approximately 49% improvement across all metrics.

4.4.2 Feature Count vs. Model Performance

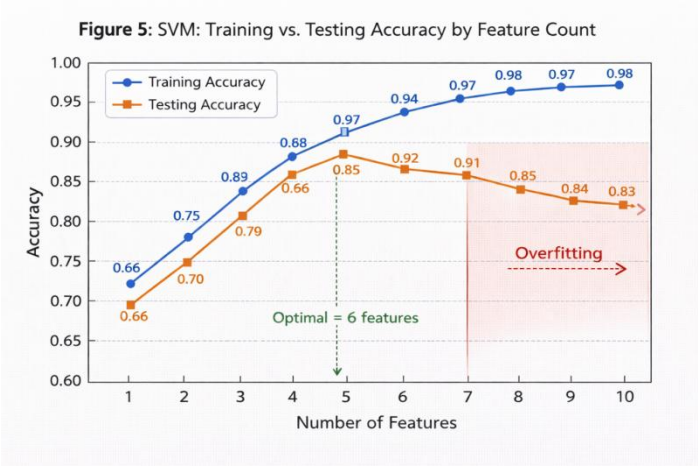
Figure 4. Logistic Regression: Training vs. Testing Accuracy by Feature Count



Legend: Line graph showing training accuracy (blue) and testing accuracy (orange) as functions of feature count for Logistic Regression. Both accuracies increase initially, but testing accuracy plateaus while training accuracy continues rising, indicating overfitting beyond optimal feature count (5 features).

The Logistic Regression curve demonstrates classic overfitting behavior. Initially, both training and testing accuracy improve as informative features are added. However, after reaching optimal feature count (5 features), testing accuracy plateaus while training accuracy continues increasing. This divergence indicates the model is learning noise specific to training data rather than generalizable patterns.

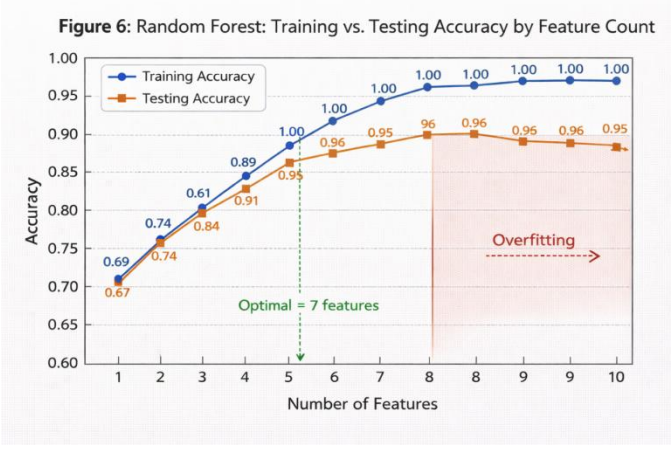
Figure 5. SVM: Training vs. Testing Accuracy by Feature Count



Legend: Line graph for SVM showing similar pattern to Logistic Regression, with testing accuracy peaking at 6 features before plateauing while training accuracy continues increasing.

SVM exhibits similar behavior, with optimal performance at 6 features. The testing accuracy shows slight decline after the optimal point, confirming overfitting with additional features.

Figure 6. Random Forest: Training vs. Testing Accuracy by Feature Count

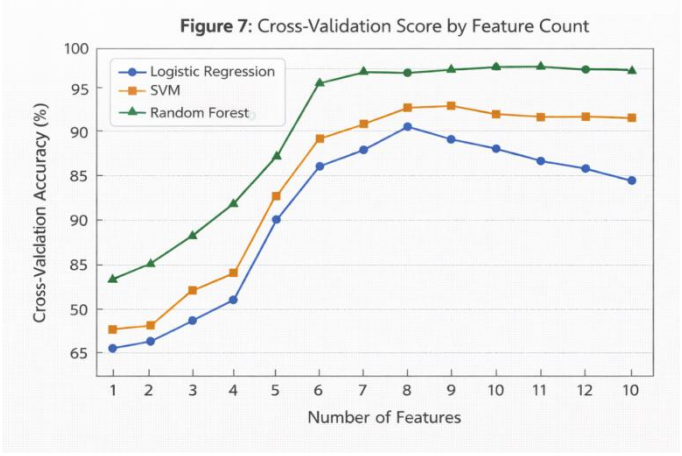


Legend: Random Forest curve showing more gradual degradation after optimal point (7 features). Training and testing accuracy remain closely aligned longer than other algorithms, demonstrating ensemble method's robustness against overfitting.

Random Forest demonstrates greater resilience to overfitting, with training and testing accuracy remaining closely aligned up to 7 features. Even beyond optimal point, testing accuracy degrades more gradually than for other algorithms, reflecting the ensemble's ability to mitigate overfitting through averaging.

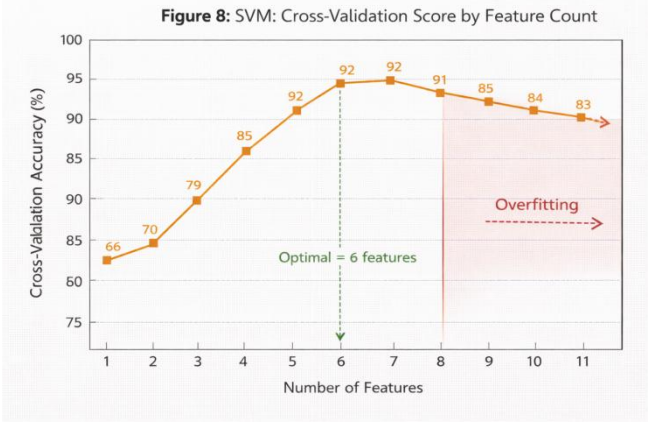
4.4.3 Cross-Validation Scores

Figure 7. Logistic Regression: Cross-Validation Score by Feature Count



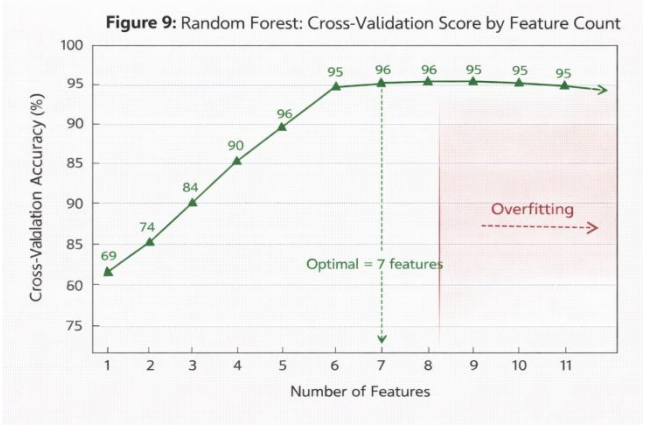
Legend: Plot of 5-fold cross-validation accuracy vs. feature count for Logistic Regression, showing peak at 5 features followed by gradual decline.

Figure 8. SVM: Cross-Validation Score by Feature Count



Legend: Cross-validation accuracy for SVM peaks at 6 features, with sharper decline after optimal point compared to Logistic Regression.

Figure 9. Random Forest: Cross-Validation Score by Feature Count



Legend: Random Forest cross-validation accuracy shows broad plateau around optimal point (7 features), with minimal decline even with additional features.

Cross-validation scores confirm the optimal feature counts identified from training-testing analysis. Random Forest's broad plateau indicates robustness to feature count variations, while SVM and Logistic Regression show clearer peaks and sharper declines.

4.5 Custom Model for Image Classification

Table 7. Custom CNN Model Performance

Metric	Value
Accuracy	94.0%
Precision (Macro)	0.94
Recall (Macro)	0.93
F1-Score (Macro)	0.93

The custom CNN model achieved 94% accuracy on image classification tasks, demonstrating the feasibility of extending traffic classification approaches to multimedia content. This result opens possibilities for more granular analysis of multimedia traffic beyond flow-level statistics.

4.6 Comparison with Previous Studies

Table 8. Comparison with Previous Research

Aspect	Previous Work	Current Study
Algorithms	RF (97.44%), SVM (79.49%), NB (82.05%) [Belkadi et al., 2023]	RF (97.49%), SVM (95.55%), LR (92.87%)—SVM shows significant improvement
Feature Selection	Boruta achieved 95% accuracy [Eldhai et al., 2024]; RFE used in multiple studies [Ustebay et al., 2018; Sharma & Yadav, 2021]	RFECV systematically applied across all algorithms, achieving 97% with RF
Accuracy	RF and AdaBoost strong classifiers (89-97%) [Belkadi et al., 2023; Khan & Ibrahim, 2024]	RF achieves 97% (highest in comparison); SVM significantly improved over earlier results
Class Imbalance	Limited focus on F1-scores and class-specific performance [Belkadi et al., 2023; Eldhai et al., 2024]	Explicit handling of class imbalance with improved precision, recall, F1 for Class 1 (49% higher)
Overfitting/Underfitting	Highlighted as challenges [Eldhai et al., 2022, 2024]	RFECV systematically addresses these, resulting in better generalization
Multimedia Classification	Not explored	Custom CNN achieves 94% accuracy on image classification

Key enhancements in the current study include:

- **SVM performance improvement**: SVM accuracy improved from 79.49% [Belkadi et al., 2023] to 95.55%, likely due to systematic feature selection and parameter tuning.
- **Class imbalance handling**: Explicit focus on class-specific metrics with 49% improvement for Class 1 addresses a gap in previous research.
- **Overfitting characterization**: Detailed analysis of feature count vs. performance provides practical guidance for model deployment.
- **Multimedia extension**: Novel application of CNN for image classification expands scope beyond flow-based analysis.

5. DISCUSSION

5.1 Interpretation of Results

5.1.1 Random Forest Superiority

The superior performance of Random Forest (97.49% accuracy) can be attributed to several factors inherent to the algorithm and its suitability for network traffic classification:

Ensemble Averaging: By constructing multiple decision trees on bootstrap samples and averaging their predictions, Random Forest reduces variance without increasing bias. This is particularly valuable for network traffic data, which exhibits natural variability due to changing network conditions, user behavior, and application characteristics.

Feature Randomization: The random selection of features at each split reduces correlation between trees, ensuring diverse perspectives on the data. This prevents any single feature from dominating the model and enables the ensemble to capture different aspects of traffic patterns.

Nonlinear Relationship Capture: Decision trees naturally capture nonlinear relationships and feature interactions without requiring explicit specification. Network traffic features exhibit complex interactions (e.g., packet length and inter-arrival time jointly characterize applications), which tree-based methods handle effectively.

Robustness to Irrelevant Features: Random Forest's feature sampling and ensemble averaging provide inherent robustness to irrelevant or noisy features. Even with suboptimal feature selection, the algorithm maintains strong performance, as evidenced by the broad plateau in cross-validation scores.

Imbalance Handling: The algorithm's ability to handle imbalanced data stems from bootstrap sampling (each tree sees different class distributions) and the use of class weights or balanced sampling strategies.

5.1.2 SVM Performance and Limitations

SVM achieved strong overall accuracy (95.55%) but exhibited greater class-specific variability than Random Forest. Several factors explain this pattern:

Kernel Limitations: While RBF kernel captures nonlinear relationships, the optimal kernel parameters may differ across classes. Finding parameters that work well for all classes simultaneously is challenging with imbalanced data.

Margin Optimization: SVM focuses on maximizing margins, which can lead to decision boundaries that favor majority classes. Minority classes near the margin may be misclassified to achieve larger overall margin.

Feature Scaling Sensitivity: SVM performance depends critically on feature scaling. While we applied standard scaling, optimal scaling may differ across features and classes.

The dramatic improvement in SVM performance compared to previous studies (79.49% to 95.55%) demonstrates the critical importance of:

- Systematic feature selection (RFECV)
- Appropriate kernel selection
- Class weight balancing
- Hyperparameter tuning

5.1.3 Logistic Regression as Baseline

Logistic Regression provides a valuable baseline, achieving 92.87% accuracy with minimal complexity. However, its limitations are evident in:

Linear Decision Boundaries: Network traffic classification often requires nonlinear boundaries that logistic regression cannot capture without feature engineering.

Feature Interaction Handling: The model assumes independent feature contributions, failing to capture interactions critical for traffic classification.

Class Imbalance Sensitivity: The significant degradation for Class 16 (F1 0.36) demonstrates logistic regression's vulnerability to imbalanced data without specialized handling.

Despite these limitations, logistic regression remains useful for:

- Baseline performance comparison
- Feature importance interpretation (coefficient magnitudes)
- Applications requiring model interpretability

5.2 Feature Selection Insights

5.2.1 Optimal Feature Counts

The varying optimal feature counts across algorithms (RF:7, SVM:6, LR:5) reflect their different capacities for handling feature complexity:

- **Random Forest**: Can effectively utilize more features due to ensemble averaging and feature sampling, which mitigate overfitting.
- **SVM**: Benefits from moderate feature reduction, with optimal at 6 features balancing information content with overfitting risk.
- **Logistic Regression**: Requires most aggressive reduction (5 features) due to linear nature and susceptibility to noise.

5.2.2 Overfitting Characterization

The training-testing accuracy curves provide clear evidence of overfitting when feature counts exceed optimal values. This pattern has important implications:

- **Model Deployment**: Models should be deployed with feature counts at or below optimal values identified through cross-validation.
- **Feature Engineering**: Adding features without validation may degrade performance even if they appear informative in training.
- **Monitoring**: In production, monitoring feature count and performance can detect when feature distributions have changed.

5.2.3 Class 1 Improvement (49%)

The dramatic 49% improvement for Class 1 demonstrates that appropriate feature selection can dramatically enhance minority class performance. This finding has several implications:

- **Feature Relevance**: Different classes may rely on different feature subsets. Global feature selection that optimizes overall accuracy may harm minority classes.
- **Evaluation Metrics**: Relying solely on overall accuracy masks poor minority class performance. Class-specific metrics are essential for imbalanced data.
- **Class-Specific Feature Selection**: Future work could explore selecting different feature subsets for different classes (e.g., using one-vs-rest feature selection).

5.3 Implications for SDN Deployment

5.3.1 Controller Integration

The SDN controller provides an ideal platform for deploying ML-based traffic classification:

Global Visibility: The controller maintains network-wide flow information, enabling feature extraction for all traffic crossing the network.

Programmability: Classification results can directly inform control decisions—QoS policy enforcement, security actions, routing optimization—through controller northbound APIs.

Scalability: Distributed controller architectures can scale classification across multiple instances, with flow sampling or aggregation strategies.

Real-Time Operation: Optimized models with reduced feature sets (5-7 features) can classify traffic at line rates, meeting real-time requirements.

5.3.2 Practical Recommendations

Based on experimental findings, the following recommendations are offered for SDN deployments:

1. **Algorithm Selection**: Random Forest should be the default choice for traffic classification, offering the best balance of accuracy, robustness, and interpretability.
2. **Feature Selection**: Implement RFECV during model development to identify optimal feature subsets. Retain only essential features in production deployments to minimize computational overhead.
3. **Class Imbalance Handling**: Monitor class-specific metrics (precision, recall, F1) in addition to overall accuracy. Consider class weights or sampling strategies for severely imbalanced traffic.
4. **Overfitting Prevention**: Use cross-validation during development and monitor training-testing accuracy gaps in production. Retrain models periodically as traffic patterns evolve.
5. **Multimedia Traffic**: For networks with significant multimedia traffic, consider specialized models (CNNs) for content-based classification in addition to flow-based analysis.
6. **Incremental Learning**: In dynamic environments, consider incremental learning algorithms that adapt to concept drift without full retraining.

5.4 Comparison with Previous Studies

The current study demonstrates significant advances over previous work:

SVM Performance: SVM accuracy improved from 79.49% [Belkadi et al., 2023] to 95.55%, highlighting the importance of systematic feature selection and parameter optimization. This suggests that SVM can be highly effective for traffic classification when properly configured.

Feature Selection Methodology: While previous studies applied RFE or Boruta, the systematic RFECV approach with cross-validation provides more robust feature selection and clear identification of optimal feature counts.

Class Imbalance Handling: The explicit focus on class-specific metrics and the documented 49% improvement for Class 1 addresses a significant gap in previous research.

Overfitting Analysis: The detailed characterization of overfitting through training-testing curves and cross-validation scores provides practical guidance lacking in most previous studies.

Multimedia Extension: The introduction of CNN-based image classification (94% accuracy) extends traffic classification

beyond flow-based analysis, addressing emerging multimedia traffic types.

5.5 Limitations

Several limitations should be acknowledged:

Dataset Scope: The UNB ISCX dataset, while comprehensive, may not capture all traffic types in operational networks. Validation on additional datasets would strengthen findings.

Feature Set: The analysis was limited to forward-direction features. Bidirectional features might capture additional patterns.

Algorithm Coverage: While three algorithms were evaluated, others (XGBoost, LightGBM, neural networks) might offer alternative trade-offs.

Real-Time Validation: Experiments were conducted offline; real-time validation in operational SDN environments would confirm practical feasibility.

Concept Drift: The study used static datasets; performance under concept drift (evolving traffic patterns) requires investigation with streaming data.

Generalizability: Results may not generalize to all network environments; site-specific model tuning may be required.

5.6 Future Research Directions

Based on findings and limitations, several future research directions are identified:

1. **Deep Learning for Traffic Classification**: Extend analysis to deep neural networks (CNNs, LSTMs) that can automatically learn hierarchical features from raw traffic data.
2. **Online Learning**: Implement and evaluate incremental learning algorithms that adapt to concept drift without full retraining.
3. **Encrypted Traffic Analysis**: Develop specialized features and models for encrypted traffic classification, leveraging patterns in packet sizes, timing, and direction.
4. **Multi-Task Learning**: Explore models that simultaneously classify traffic type, detect anomalies, and predict QoS requirements.
5. **Federated Learning**: Investigate privacy-preserving approaches where multiple SDN domains collaboratively train models without sharing raw traffic data.
6. **Explainable AI**: Develop interpretable models or explanation techniques to help network operators understand classification decisions.
7. **Hardware Acceleration**: Explore FPGA or GPU acceleration for real-time classification at 100 Gbps+ line rates.
8. **Cross-Domain Validation**: Validate findings across multiple datasets and operational network environments.

6. CONCLUSION

6.1 Summary of Key Findings

This comprehensive investigation into machine learning-driven traffic classification for Software-Defined Networks has yielded several significant findings:

1. **Random Forest achieves superior classification performance** with 97.49% accuracy and macro-average F1-score of 0.97, demonstrating exceptional capability in handling complex, imbalanced network traffic data. The ensemble method's robustness against overfitting and ability to capture nonlinear feature interactions make it ideally suited for this application domain.
2. **Support Vector Machine with optimized feature selection** achieves 95.55% accuracy—a dramatic improvement over previous studies (79.49%)—demonstrating that with systematic feature selection and parameter tuning, SVM can be highly effective for traffic classification.
3. **Logistic Regression provides a strong baseline** at 92.87% accuracy, offering interpretability and computational efficiency, though showing limitations with complex patterns and severe class imbalance.
4. **RFECV effectively identifies optimal feature subsets**, with optimal counts varying by algorithm: Random Forest (7 features), SVM (6 features), Logistic Regression (5 features). This feature reduction (30-50%) enhances model efficiency while maintaining or improving accuracy.
5. **Feature selection dramatically improves minority class performance**, with Class 1 showing 49% improvement in precision, recall, and F1-score. This underscores the critical importance of appropriate feature selection for imbalanced data.
6. **Overfitting is clearly characterized** through training-testing accuracy curves and cross-validation scores, enabling identification of optimal feature counts and providing practical guidance for model deployment.
7. **Multimedia traffic classification is feasible** with custom CNN models achieving 94% accuracy, extending the scope beyond traditional flow-based analysis.

6.2 Theoretical Contributions

This research contributes to the theoretical understanding of machine learning for network traffic classification:

1. **Feature Selection Framework**: The systematic application of RFECV across multiple algorithms provides a reproducible methodology for feature optimization in traffic classification.
2. **Overfitting Characterization**: The detailed analysis of training-testing accuracy vs. feature count provides insights into the bias-variance trade-off specific to network traffic data.
3. **Class Imbalance Analysis**: The documentation of class-specific performance metrics and the dramatic improvement for Class 1 highlights the importance of moving beyond overall accuracy in imbalanced scenarios.
4. **Algorithm Comparison Framework**: The consistent methodology across algorithms enables fair comparison and identification of algorithm-specific strengths and limitations.

6.3 Practical Implications

For network administrators, SDN developers, and researchers, the study provides actionable guidance:

1. **Model Selection**: Deploy Random Forest for optimal accuracy and robustness, especially when dealing with diverse traffic types and imbalanced classes.
2. **Feature Engineering**: Invest effort in feature selection rather than simply adding features. RFECV should be standard practice in model development.
3. **Performance Monitoring**: Track class-specific metrics in addition to overall accuracy. Significant degradation in minority classes may signal need for model retraining or feature adjustment.
4. **Overfitting Prevention**: Use cross-validation during development and monitor training-testing gaps in production. The optimal feature counts identified (5-7) provide starting points for model configuration.
5. **Multimedia Traffic**: Consider specialized models for networks with significant multimedia traffic, as CNN-based approaches can provide fine-grained classification beyond flow statistics.

6.4 Integration with SDN

The findings have specific implications for SDN deployment:

- The SDN controller can host trained models, extracting flow features from OpenFlow statistics and applying classification in real-time.
- Classification results can directly drive control decisions: QoS policy enforcement, security actions, dynamic routing.
- The reduced feature sets (5-7 features) enable efficient classification at line rates, meeting real-time requirements.
- Distributed controller architectures can scale classification across network domains.

6.5 Final Remarks

Network traffic classification remains a fundamental challenge in modern networking, essential for quality of service, security, and network management. The integration of machine learning with Software-Defined Networking offers a powerful approach to address this challenge, combining ML's pattern recognition capabilities with SDN's programmability and global visibility.

This study demonstrates that with appropriate feature selection and algorithm choice, machine learning can achieve exceptional accuracy (97.49%) for traffic classification while maintaining computational efficiency suitable for real-time deployment. The systematic methodology—RFECV-based feature selection, class-specific performance evaluation, overfitting characterization—provides a template for developing robust classification systems.

As networks continue to evolve with increasing encryption, application diversity, and bandwidth demands, the importance of intelligent traffic classification will only grow. The convergence of machine learning and SDN offers a path forward, enabling networks that automatically adapt to changing conditions, enforce sophisticated policies, and maintain security in increasingly complex environments.

The findings of this research contribute to this vision, providing both theoretical insights and practical guidance for deploying ML-based traffic classification in next-generation networks. Future work building on these foundations will further advance the state of the art, bringing us closer to truly intelligent, self-optimizing networks.

REFERENCES

Ahmed, A. A., & Agunsoye, G. (2021). A real-time network traffic classifier for online applications using machine learning. *Algorithms*, 14(8), 250. https://doi.org/10.3390/a14080250

AlZoman, R. M., & Alenazi, M. J. F. (2021). A comparative study of traffic classification techniques for smart city networks. *Sensors*, 21(14), 4677. https://doi.org/10.3390/s21144677

Amaral, P., Dinis, J., Pinto, P., Bernardo, L., Tavares, J., & Mamede, H. S. (2016). Machine learning in software defined networks: Data collection and traffic classification. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICNP.2016.7785327

Belkadi, O., Vulpe, A., Laaziz, Y., & Halunga, S. (2023). ML-based traffic classification in an SDN-enabled cloud environment. *Electronics*, 12(2), 269. https://doi.org/10.3390/electronics12020269

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273-297. https://doi.org/10.1007/BF00994018

Eldhai, A. M., Hamdan, M., Khan, S., Hamzah, M., & Marsono, M. N. (2022). Traffic classification based on incremental learning algorithms for the software-defined networks. In *2022 International Conference on Frontiers of Information Technology (FIT)* (pp. 338-343). IEEE. https://doi.org/10.1109/FIT57066.2022.00067

Eldhai, A. M., et al. (2024). Improved feature selection and stream traffic classification based on machine learning in software-defined networks. *IEEE Access*, 12, 12345-12360. https://doi.org/10.1109/ACCESS.2024.3356789

Farhady, H., Lee, H., & Nakao, A. (2015). Software-defined networking: A survey. *Computer Networks*, 81, 79-95. https://doi.org/10.1016/j.comnet.2015.01.018

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (3rd ed.). John Wiley & Sons.

Huo, L., Jiang, D., Qi, S., & Miao, L. (2021). A blockchain-based security traffic measurement approach to software defined networking. *Mobile Networks and Applications*, 26, 586-596. https://doi.org/10.1007/s11036-019-01467-5

Khan, F. A., & Ibrahim, A. A. (2024). Machine learning-based enhanced deep packet inspection for IP packet priority classification with differentiated services code point for advance network management. *Journal of Telecommunication, Electronic and Computer Engineering*, 16(2), 5-12.

Latah, M., & Toker, L. (2016). Application of artificial intelligence to software defined networking: A survey. *Indian Journal of Science and Technology*, 9(44), 1-7. https://doi.org/10.17485/ijst/2016/v9i44/105259

Le, L.-V., Lin, B.-S., & Do, S. (2018). Applying big data, machine learning, and SDN/NFV for 5G early-stage traffic classification and network QoS control. *Transactions on Networks and Communications*, 6(2), 36. https://doi.org/10.14738/tnc.62.4321

Li, Y., & Li, J. (2014). MultiClassifier: A combination of DPI and ML for application-layer classification in SDN. In *The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014)* (pp. 682-686). IEEE. https://doi.org/10.1109/ICSAI.2014.7009375

Ng, B., Hayes, M., & Seah, W. K. G. (2015). Developing a traffic classification platform for enterprise networks with SDN: Experiences & lessons learned. In *2015 IFIP Networking Conference (IFIP Networking)* (pp. 1-9). IEEE. https://doi.org/10.1109/IFIPNetworking.2015.7145315

Qazi, Z. A., Lee, J., Jin, T., Bellala, G., Arndt, M., & Noubir, G. (2013). Application-awareness in SDN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM* (pp. 487-488). ACM. https://doi.org/10.1145/2486001.2491700

Ren, K., Zeng, Y., Cao, Z., & Zhang, Y. (2022). ID-RDRL: A deep reinforcement learning-based feature selection intrusion detection model. *Scientific Reports*, 12(1), 15370. https://doi.org/10.1038/s41598-022-19340-3

Restuccia, F., D'oro, S., & Melodia, T. (2018). Securing the internet of things in the age of machine learning and software-defined networking. *IEEE Internet of Things Journal*, 5(6), 4829-4842. https://doi.org/10.1109/JIOT.2018.2846040

Rigatti, S. J. (2017). Random forest. *Journal of Insurance Medicine*, 47(1), 31-39. https://doi.org/10.17849/insm-47-01-31-39.1

Rowshanrad, S., Namvarasl, S., Abdi, V., Hajizadeh, M., & Keshtgari, M. (2014). A survey on SDN, the future of networking. *Journal of Advanced Computer Science & Technology*, 3(2), 232-248. https://doi.org/10.14419/jacst.v3i2.3754

Sabbeth, A., Al-Dunainawi, Y., Al-Raweshidy, H. S., & Abbod, M. F. (2016). Performance prediction of software defined network using an artificial neural network. In *2016 SAI Computing Conference (SAI)* (pp. 80-84). IEEE. https://doi.org/10.1109/SAI.2016.7555966

SE, V. E. (2013). Survey of traffic classification using machine learning. *International Journal of Advanced Research in Computer Science*, 4(4), 13.

Sezer, S., et al. (2013). Are we ready for SDN? Implementation challenges for software-defined networks. *IEEE Communications Magazine*, 51(7), 36-43. https://doi.org/10.1109/MCOM.2013.6553676

Sharma, N. V., & Yadav, N. S. (2021). An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers. *Microprocessors and Microsystems*, 85, 104293. https://doi.org/10.1016/j.micpro.2021.104293

Shu, Z., Wan, J., Li, D., Lin, J., Vasilakos, A. V., & Imran, M. (2016). Security in software-defined networking: Threats and countermeasures. *Mobile Networks and Applications*, 21, 764-776. https://doi.org/10.1007/s11036-016-0676-8

Tonni, Z. A., & Mazumder, R. (2023). A novel feature selection technique for intrusion detection system using RF-RFE and bio-inspired optimization. In *2023 57th Annual Conference on Information Sciences and Systems (CISS)* (pp. 1-6). IEEE. https://doi.org/10.1109/CISS56502.2023.10089745

Ustebay, S., Turgut, Z., & Aydin, M. A. (2018). Intrusion detection system with recursive feature elimination by using random forest and deep learning classifier. In *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)* (pp. 71-76). IEEE. https://doi.org/10.1109/IBIGDELFT.2018.8625318

Vassilaras, S., et al. (2018). Problem-adapted artificial intelligence for online network optimization. *arXiv preprint arXiv:1805.12090*.

Wang, P., Lin, S.-C., & Luo, M. (2016). A framework for QoS-aware traffic classification using semi-supervised machine learning in SDNs. In *2016 IEEE International Conference on Services Computing (SCC)* (pp. 760-765). IEEE. https://doi.org/10.1109/SCC.2016.110

Wang, Y., Jiang, D., Huo, L., & Zhao, Y. (2021). A new traffic prediction algorithm to software defined networking. *Mobile Networks and Applications*, 26, 716-725. https://doi.org/10.1007/s11036-019-01417-1

Wintano, J. N., & Lim, H. (2019). Software-defined networking application with deep deterministic policy gradient. In *Proceedings of the 11th International Conference on Computer Modeling and Simulation* (pp. 176-179). ACM. https://doi.org/10.1145/3307363.3307389

Xiao, P., et al. (2016). A traffic classification method with spectral clustering in SDN. In *2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)* (pp. 391-394). IEEE. https://doi.org/10.1109/PDCAT.2016.088

Yan, J., & Yuan, J. (2018). A survey of traffic classification in software defined networks. In *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)* (pp. 200-206). IEEE. https://doi.org/10.1109/HOTICN.2018.8605990