International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 18s, 2025
https://theaspd.com/index.php

Improving Software-Defined Network Efficiency With Random Forest Model

Saroj Singh[1], Dr. Kamlesh Sharma[2]
[1]Department of CSE SET, Manav Rachna International Institute of Research and Studies, Faridabad - 121004, Haryana, India, sarojraj47@gmail.com
[2]Department of CSE SET, Manav Rachna International Institute of Research and Studies, Faridabad - 121004, Haryana, India, associatedean_ks.academics@mriu.edu.in

Abstract
The increasing demand for real-time services, data-intensive applications, and scalable network infrastructures has challenged traditional networking models, highlighting the need for intelligent, adaptive solutions. Software-Defined Networking (SDN) addresses this by separating the control and data planes, allowing for centralized and programmable network management. However, existing SDN systems still struggle with issues such as latency, controller bottlenecks, and inefficient traffic handling under dynamic conditions. This study presents a novel approach to enhancing SDN efficiency by integrating a Bayesian-optimized Random Forest (RF) model within the controller framework. The proposed methodology involves detailed data preprocessing, feature selection using Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA), and model training with hyperparameter tuning via Bayesian optimization. Real-time integration of the RF model enables intelligent flow classification and decision-making within the SDN controller. Experimental results indicate a dramatic improvement in throughput, lower latency, and the total removal of packet loss compared to the baseline. The model reached a maximum accuracy of 99.99%, outperforming existing solutions and revealing the potential of ensemble learning in contemporary network environments. The study contributes to a solid and practical approach to designing intelligent, data-driven SDN systems.

Keywords: *Software-Defined Networking, Random Forest, Flow Classification, Bayesian Optimization, Network Efficiency*

1. INTRODUCTION

The rapid growth in data-driven services, real-time applications, and cloud-based platforms has put enormous stress on conventional network infrastructures, requiring innovative and responsive networking solutions [1,2]. With digital ecosystems becoming highly dynamic and distributed, traditional network architectures with fixed hardware-based control planes are found to be lacking in scalability, flexibility, and real-time responsiveness requirements [3-5]. In such an environment, Software-Defined Networking (SDN) has gained prominence as a new networking model, offering a distinct design by separating the control plane from the data plane [6-8]. This decoupling enables centralized network control, dynamic configuration, and enhanced visibility throughout the network, allowing network administrators to optimize performance and enforce policies with enhanced accuracy [9,10].

While SDN architecture has some advantages, it also presents challenges in scalability, robustness of the controller, latency, security, interoperability, and management complexity. There would be a situation of bottleneck for the centralized controller when there is high traffic or large-scale networks, which leads to high latency, improper routing decisions, flow table overflows, and low reactivity [11-13]. Apart from that, the dynamic and complex nature of current networks, which are caused by shifting traffic patterns, security attacks, and heterogeneous devices, necessitates smart, context-aware decision-making processes that are beyond fixed rule-based control logic [14-16]. Automation and intelligence in SDNs have also generated increasing interest in integrating Machine Learning (ML) techniques to enhance different network operations such as traffic classification, anomaly detection, and routing optimization [17]. Figure 1 illustrates the overall architecture of SDN.

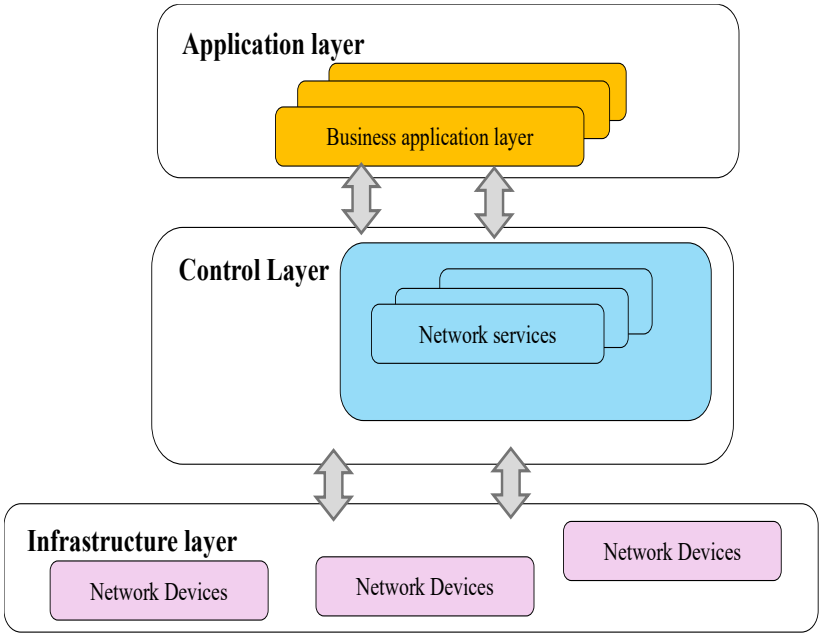

Figure 1: Software Defined Network Architecture [18]

Among all available ML algorithms, the RF model has remained popular because of its scalability, robustness, and high classification accuracy [19,20]. As an ensemble learning technique based on decision trees, RF attains high predictive accuracy, resistance to overfitting, and high efficiency when processing high-dimensional data and noisy inputs, attributes highly sought after in real-time network situations [21-23]. For SDN cases, RF can efficiently classify traffic flows, predict optimal paths, detect anomalous activities, and support load balancing while maintaining low CPU overhead [24-26].

This study aims to enhance the efficiency and responsiveness of SDNs by integrating an RF model into the controller's decision-making framework. The focus is on enabling intelligent flow classification and real-time traffic prediction to optimize network performance. The scope includes the proposed model's design, implementation, and empirical evaluation using benchmark datasets that simulate real-world traffic patterns and controller loads. Through this, the study investigates how the RF model can reduce latency, improve throughput, and enhance the overall adaptability of SDNs under varying traffic conditions. The significance of this research lies in its contribution to the development of more intelligent and more autonomous network management systems. The study demonstrates how machine learning can transition SDNs from rule-based management to predictive and data-driven operation by applying RF to the SDN environment. The key contributions include:

- Proposing a novel integration of RF for flow classification and traffic prediction within the SDN control plane.
- Employing Bayesian optimization to tune RF hyperparameters, achieving peak F1-score with minimal evaluation trials.
- Engineering an SDN-specific preprocessing pipeline by including median imputation, winsorization, one-hot encoding, and derived metrics to capture key traffic patterns.
- Validating significant throughput, latency, and packet loss improvements through testbed experiments and one-way Analysis of Variance (ANOVA).

This research bridges theoretical advancements with practical SDN applications, laying the groundwork for further study in utilizing ensemble learning models to build self-optimizing and intelligent networks. The sections are structured as follows: Section 1 presents background knowledge, including an introduction to the research field, and Section 2 summarizes several previous studies conducted by various authors. Section 3 details the proposed methodology adopted in this research. Section 4 discusses the experimental results and provides an in-depth analysis of the model's performance across key metrics. Finally, Section 5 concludes the study with a summary of key findings and outlines potential directions for future research.

2. LITERATURE REVIEW

This literature review surveys key studies on machine-learning and optimization approaches for traffic classification, resource prioritization, and performance enhancement in SDN environments. In 2025, Shahgholi et al. [27] introduced an active reward learning framework within a deep reinforcement learning agent to optimize resource allocation for 5G network slicing and Intelligent Transportation Systems (ITS). By combining a Gaussian process surrogate for the reward function with strategic expert queries, their method cut average packet delay by 15%. It boosted spectrum efficiency by 10% in 5G slicing, while in ITS scenarios, it delivered a 20% drop in vehicle waiting times and a 5% increase in traffic throughput. In the same year, Serag et al. (2025) [28] proposed an SDN-based traffic classification approach using various machine learning models. The Extreme Gradient Boosting (XGBoost) classifier achieved a peak accuracy of 99.97%, outperforming both standalone and hybrid baselines with an execution time of just 3.11 seconds. Throughout 2024, Salau and Beyene [29] employed supervised and unsupervised techniques within an SDN controller to classify Domain Name Server (DNS), Telnet, Ping, and Voice flows, finding that their Decision Tree delivered the highest accuracy of 99.81% in both offline and real-time evaluations. Previously, Arif et al. (2024) [30] developed Deep Reinforcement Learning (DRL) Quality of Service (QoS) and Quality of Experience (QoE) (DQQS), a deep reinforcement learning model for secure routing in SDN-IoT environments; in a simulated attacked network it achieved throughput of 14.5 Mega Byte Per Second (Mbps) surpassing OSPF (8 Mbps), L-L (8.2 Mbps), Sailfish (9 Mbps), and RL (9.5 Mbps) and reduced latency to 52 ms versus 72–88 ms for the other protocols. In 2023, Belgaum et al. [31] introduced Self-Socio Adaptive Reliable Particle Swarm Optimization (SSAR-PSO), a self-socio adaptive, reliable PSO algorithm for SDN load balancing. Under TCP traffic, it reduced latency by up to 16.95%, improved throughput by up to 6.32%, and cut packet loss by up to 31.62% compared to PSO, Ant Colony Optimization (ACO), and Round Robin baselines. In the same year, Çavdar and Aymaz [32] proposed a discrete PSO with a hybrid cost function that balances link and switch loads in fat-tree topologies, achieving an average throughput of 158.47 Mbps, which is seven times higher than competing methods and requires just 1.1 KB of buffer, compared to 3.4–7.2 KB for others. In 2022, Jiménez-Lázaro et al. [33] applied Logistic Regression (LR-EE) to predict energy-efficient SDN configurations. With 200 clusters, LR-EE outperformed a Genetic Algorithm by saving an average of 6.95% more power, maintaining over 95% feasibility, and accelerating prediction times by factors ranging from 526,190× to 1,473,333×. However, Kumar et al. (2022) [34] developed a CNN-based SDN load-balancer that achieved 98.94% training accuracy and 99.22% validation accuracy with only 3.61% loss, demonstrating high efficacy and fast convergence. Finally, in 2021, Alhaidari et al. [35] presented an Intelligent SDN framework using Deep Extreme Learning Machines for cognitive routing optimization; while they reported "superior results" in convergence and route quality, quantitative gains were not explicitly specified.

Despite substantial advances, key gaps persist. Many authors relied primarily on benchmark sets like IoT-23 and NSL-KDD, leaving their methods untested on larger, more heterogeneous modern traffic traces [30]. The logistic regression approach demonstrated up to 6.95% additional power savings over genetic algorithms; however, its scalability to large-scale topologies and real-time controller workflows remains unexamined [33]. In contrast, Active Reward Learning reduced average packet delay by 15% and improved spectrum efficiency by 10%, all while maintaining surrogate accuracy under rapidly changing 5G conditions [27].

3. RESEARCH METHODOLOGY

In this section, the proposed methodology is described in detail. The data is taken from Kaggle and then cleaned with median imputation, outliers of extreme values, and one-hot encoding of categorical fields. Further metrics such as bytes per packet, flows per second, and time-of-day features are engineered. The dataset was then stratified into 70% training and 30% test sets. An RF was trained, and its tree count, depth, and leaf size were tuned via Bayesian optimization. Efficiency gains were measured by comparing throughput, latency, and packet loss between the baseline and RF-driven setups, and these differences were confirmed to be significant using one-way ANOVA. A systematic flowchart of the proposed methodology is provided in Figure 2.

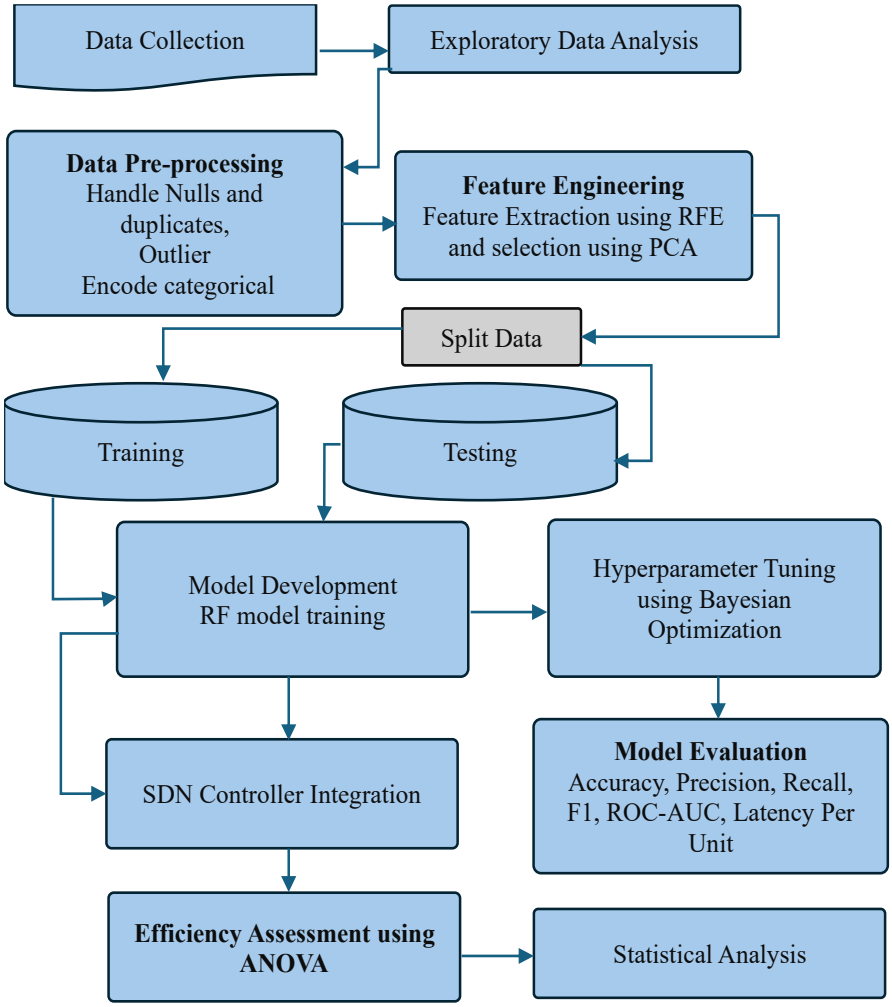

Figure 2: Systematic Flowchart of the proposed methodology

3.1Data Collection

The dataset consists of 104345 one-second flow and port statistics snapshots polled from an OpenFlow-enabled SDN testbed [36]. It includes timestamp (dt), switch and port identifiers (switch, port_no), flow endpoints (src, dst, Protocol), raw counters (pktcount, bytecount, dur, dur_nsec, tot_dur, flows, packetins), derived metrics (pktperflow, byteperflow, pktrate, Pairflow), throughput measurements (tx_bytes, rx_bytes, tx_kbps, rx_kbps, tot_kbps), and a binary label (0 benign, one malicious). This combination of temporal, topological, volumetric, and performance features provides a comprehensive view of network behavior for training and evaluating traffic-prioritization models.

3.2Exploratory Data Analysis

Exploratory Data Analysis [37] is a critical phase in the data-driven pipeline, aimed at understanding the structure, quality, and patterns within the raw OpenFlow controller data collected from the Software-Defined Network (SDN) environment [38]. Exploratory Data Analysis began by computing descriptive statistics for each flow-level metric to reveal central tendencies and dispersion. Histograms, box plots, and kernel density estimates were then plotted to identify outliers and non-normal distributions, which informed subsequent winsorization thresholds. A Pearson correlation heatmap revealed multicollinearity among the features, guiding the removal of redundant variables before feature selection. Temporal plots of flow statistics over hourly intervals uncovered diurnal traffic patterns, while label-balance checks on benign versus malicious flows ensured that stratified sampling would preserve class proportions. Finally, a completeness audit located missing entries, confirming data integrity and setting the stage for robust preprocessing and feature engineering.

3.3Data Preprocessing

Data preprocessing began by loading the OpenFlow statistics into a tabular DataFrame and checking each column for missing or inconsistent entries. Null values were filled with median estimates, and any duplicate records or flows reporting zero packets or zero duration were removed to prevent invalid calculations. Numeric features displaying extreme skew were winsorized to tame outliers, and all

categorical fields, such as protocol type and switch identifier, were converted into one-hot encoded vectors. This clean, uniform dataset formed the foundation for feature engineering and model development.

3.4Feature Engineering

- **RFE for Feature Extraction**

Recursive feature elimination is a straightforward method for winnowing down a large set of predictors to a handful that matter [39,40]. Starting with all candidate variables, it fits a model for data solving:

$$w = \arg\min_{w} L(Xw, y) \quad (1)$$

For a chosen loss function L, it assigns each feature i an importance score:

$$I_i = | w_i | \quad (2)$$

The feature with the smallest I_i is dropped, and the process repeats on the reduced set until only the desired number of predictors remains. By successively culling the weakest variables, RFE yields a compact bundle of features that drive the best performance. It is applied to rank all flow-level and port-level features by importance, successively removing the least influential variable at each iteration. This iterative pruning produced a concise set of predictors that fed into the RF model, improving training speed and classification reliability.

- **PCA for Feature Selection**

Principal component analysis seeks to establish a new coordinate system in which the first axes capture as much of the data's variance as possible [41,42]. After centering the data matrix X, one forms the covariance:

$$\sum = \frac{1}{n-1} X^T X \quad (3)$$

And solves the eigenvalue problem:

$$\Sigma v = \lambda v \quad (4)$$

The Component and its corresponding eigenvalue λ measure the variance in that direction. Ordering these components by descending λ and projecting X onto the top k eigenvectors produces a lower-dimensional representation that retains the bulk of the original information while filtering out noise. PCA was used to transform highly correlated throughput and duration metrics into orthogonal components, retaining only the top three principal axes of the total variance proposed by the methodology. By projecting the data onto these dimensions, noise and dimensionality are reduced while preserving the core traffic patterns needed for accurate flow prioritisation. After performing feature engineering, the data were partitioned into a training and testing set using stratified sampling on the label to preserve class balance.

3.5Model Development: RF Model

RF modeling began by growing an ensemble of B decision trees, each trained on a different bootstrap sample of the training data and, at every split, considering a random subset of features [43,44]. For a new flow feature vector x, each tree b casts a vote fb(x), and the forest's final decision is the most frequent class among all trees:

$$\hat{y} = \arg\max_{c} \sum_{b=1}^{B} I(f_b(x) = c) \quad (5)$$

Where 'I' is the indicator function. Key parameters such as the number of trees B, maximum tree depth, and minimum samples per leaf were tuned via randomized search to maximize the F1-score on the validation set. In the proposed methodology, this bagging-and-feature-randomization approach reduces overfitting, improves robustness to noisy measurements, and delivers stable flow-priority predictions under varied network conditions.

3.6Hyperparameter Tuning: Bayesian Optimization

Bayesian optimization treated hyperparameter tuning as a sequential search guided by a surrogate model, specifically a Gaussian process that predicted the validation F1-score $f(\theta)$ for any combination θ of parameters [45,46]. At each step, the algorithm chose the next candidate θ by maximizing the Expected Improvement (EI) acquisition function:

$$EI(\theta) = E[\max(0, f(\theta) - f(\theta^+))], \quad (6)$$

Where $f(\theta^+)$ denotes the best F1-score observed so far, and the expectation is taken over the Gaussian posterior at θ. After evaluating the RF with those settings, the new result updated the Gaussian process, refining its predictions. This loop continued until improvement plateaued, yielding tuned values for the number of trees, tree depth, and leaf size that maximized classification performance with minimal trial runs.

Pseudo Code

Inputs:

- f(θ): Validation F1-score as a function of hyperparameters θ
- Θ: Search space for θ
- n_initial: Number of initial samples
- N_iterations: Total number of trials

1.Generate Initial Samples

Sample $\theta_1,\ldots,\theta_n$_initial uniformly from Θ

For each θ_i, compute $y_i = f(\theta_i)$

2. Fit Surrogate Model

Use $\{(\theta_i, y_i)\}$ to train a Gaussian Process (GP) that predicts f and its uncertainty

3. Iterative Optimization

For t = n_initial + 1 to N_iterations:

Define Expected Improvement acquisition:

$EI(\theta) = E[\max(0, f(\theta) - y\_best)]$

Where $y\_best = \max\{y_1,\ldots,y_{t-1}\}$

Select the next hyperparameters:

$\theta\_t = \arg\max\{\theta \in \Theta\} EI(\theta)$

Evaluate performance:

$y\_t = f(\theta\_t)$

Augment data:

Add (θ_t, y_t) to the training set of the GP

Update GP surrogate with the new data

4. Return Best Setting

$\theta^* = \arg\max\{i=1\ldots N\_iterations\} y_i$

3.7SDN Controller

An SDN controller serves as the network's brain, periodically polling each switch for flow and port statistics via OpenFlow requests and translating those raw counters into the feature vector $x(t)$ [47,48]. Formally, at each interval t, the controller performs

$$x(t) = \phi(S(t)) \quad (7)$$

Where S(t) is the set of all switch-reported metrics (packet counts, byte counts, durations, throughput), and ϕ is the feature-extraction mapping. The RF model then produces a label:

$$\hat{y}(t) = f_{RF}(x(t)), \quad (8)$$

And the controller applies the decision rule.

$$a(t) = \pi(\hat{y}(t)) = \begin{cases} \text{DROP\_FLOW}, & \hat{y}(t) = 1 \\ \text{ALLOW\_FLOW}, & \hat{y}(t) = 0 \end{cases} \quad (9)$$

Installing or withholding flow-modification rules accordingly. This closed-loop integration ensures that each flow is evaluated and prioritized in real-time, allowing for optimal bandwidth utilization and rapid mitigation of problematic traffic.

3.8ANOVA for Efficiency Assessment

One-way ANOVA was used to determine whether the mean values of key performance indicators, such as throughput, latency, and packet loss, differed significantly across the test scenarios [49,50]. ANOVA decomposes the total variability into "between-group" and "within-group" components and computes the ratio of their mean squares:

$$F = \frac{MS_{between}}{MSwithin} = \frac{SSbetween/(k-1)}{SSwithin/(N-k),} \quad (10)$$

where k is the number of scenarios, N is the total number of runs, SS denotes sums of squares, and MS is the corresponding mean squares. A high F value (with $p<0.05$) indicates that at least one scenario's mean KPI differs from the others, prompting post hoc tests (e.g.\ Tukey's HSD) to pinpoint which pairs are significantly different.

3.9Proposed Algorithm

In this section, the proposed algorithm for the proposed methodology is provided.

Algorithm 1: Mathematical Algorithm for SDN Efficiency Enhancement

1. **Data Collection**

For each polling instant $t = 1, \ldots, T$, obtain raw OpenFlow metrics

$$S(t) = \{ \text{pktcount}_i(t), \text{bytecount}_i(t), \text{dur}_i(t), \ldots \}_{i=1}^{M}$$

2. **Exploratory Data Analysis (EDA)**
   - Compute descriptive statistics for all flow-level metrics.
   - Assess label balance between benign and malicious flows.
   - Identify missing entries and assess data completeness.
3. **Feature Mapping**

Apply the extraction mapping ϕ\phi$\phi$ to form feature vectors:

$$x(t) = \phi(S(t)) \in R^d$$

Where ϕ includes derived metrics such as: $\text{bytes\_per\_pkt} = \frac{\text{bytecount}}{\text{pktcount}}$ and $\text{flows\_per\_sec} = \frac{\text{flows}}{\text{dur+dur\_nsec}\times 10^{-9}}$.

4. **Data Preprocessing**
   - **Handling Missing Values**: For each feature j, if missing, then $x_j \leftarrow \text{median}(x_j)$
   - **Remove Duplicates**: Discard samples with $\text{pktcount} = 0 \text{ or } \text{dur} = 0$.
   - **Outliers**: Clip x_j to the percentile range.
   - **Encoding**: One-hot encode all categorical entries (e.g., Protocol, switch ID).
5. **Feature Engineering**
   - **RFE**:
     - Fit linear model $w = \arg\min_w L(Xw, y)$
     - Compute feature importances $I_i = | w_i |$
     - Remove the feature with the smallest. I_i; repeat until d_{RFE} remain.
   - **PCA**:
     - Center data $X \in R^{N\times dRFE}$, compute $\Sigma = \frac{1}{N-1} X^T X$
     - Solve $\Sigma v = \lambda v$, select top p eigenvectors. V_p with $\sum_{i=1}^{p} \lambda_i / \sum_i \lambda_i > 0.95$
     - Project: $Z = X V_p \in R^{N\times p}$.
6. **Data Split**

Perform stratified sampling on labels y to obtain: $\{X_{train}, y_{train}\}$, $\{X_{test}, y_{test}\}$ with ratios 70% and 30%.

7. **RF Modeling**
   - Train B trees on bootstrap samples. Each tree b yields a vote $f_b(x)$.
   - Aggregate by majority: $\hat{y}(x) = \arg\max_{c\in\{0,1\}} \sum_{b=1}^{B} I(f_b(x) = c)$.
8. **Hyperparameter Tuning (Bayesian Optimization)**
   - Define surrogate GP over $\theta = (B, \text{depth}, \text{leaf})$ to predict $f(\theta)$.
   - Iteratively select: $\theta_t = \arg\max_\theta EI(\theta)$; where $EI(\theta) = E[\max(0, f(\theta) - f^+)]$
   - Update GP with each new $(\theta_t, f(\theta_t))$ until convergence to θ^*.
9. **Controller Integration**
   - At each t, compute $\hat{y}(t) = f_{RF}(x(t))$.

Apply action: $a(t) = \begin{cases} \text{DROP\_FLOW}, & \hat{y}(t) = 1 \\ \text{ALLOW\_FLOW}, & \hat{y}(t) = 0 \end{cases}$.

10. **Performance Evaluation**
    - Measure KPIs {throughput, latency, loss} under each scenario (baseline vs. RF-driven).
11. **ANOVA Testing**
    - Compute sums of squares: $SS_{between}$ and SS_{within} .
    - Calculate: $F = SS_{between}(k-1)$.
    - If $p<0.05$, conclude that at least one scenario differs significantly.

This algorithmic framework ensures a mathematically grounded, end-to-end path from raw SDN metrics to statistically validated efficiency improvements.

4. RESULTS AND DISCUSSION

This section presents the outcomes of the proposed Random Forest-based model for improving SDN efficiency. The results demonstrate the effectiveness of feature selection and dimensionality reduction techniques, as well as the impact of Bayesian-tuned hyperparameters on model performance. Real-time integration into the SDN controller enabled smarter flow decisions, resulting in reduced network latency

and packet loss. Comparative evaluations confirm improved classification accuracy and responsiveness compared to the baseline approach.

4.1Hyperparameter

Table 1 highlights the key settings used to fine-tune the Random Forest model applied in the SDN framework. These parameters were carefully adjusted using Bayesian Optimization to improve the model's accuracy, responsiveness, and overall network performance. Each entry represents the typical value found effective during experimentation.

Table 1: Key hyperparameters used for Random Forest model tuning.

Hyperparameter	Range
Number of Trees	175
Maximum Tree Depth	20
Minimum Samples per Leaf	4
Bayesian Optimization Runs	25

4.2Evaluation Metrics

The effectiveness of the Random Forest model in the SDN setup was measured using several performance metrics. These metrics help in understanding how well the model makes decisions and how it impacts overall network performance.

Accuracy: Represents the proportion of correctly predicted outcomes among the total number of cases evaluated.

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{T P+TN+FP+FN}} \quad (11)$$

Where

TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

Precision: Measures the percentage of true positive predictions among all positive predictions made by the model.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (12)$$

Recall: Indicates how many actual positive instances were correctly identified. It reflects the model's sensitivity.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (13)$$

F1-Score: Provides a balanced average between precision and recall, especially useful in cases of imbalanced classes.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision+Recall}} \quad (14)$$

4.3Result analysis

This section presents the outcomes of the proposed Random Forest-based model for improving SDN efficiency. The results demonstrate the effectiveness of feature selection and dimensionality reduction techniques, as well as the impact of Bayesian-tuned hyperparameters on model performance. Real-time integration into the SDN controller enabled smarter flow decisions, resulting in reduced network latency and packet loss. Comparative evaluations confirm improved classification accuracy and responsiveness compared to the baseline approach.

4.3.1 Data Preprocessing

The correlation heatmap, illustrated in Figure 3 the relationships between features in the dataset, with color intensity indicating the strength of correlation. Strong positive correlations are observed between features like pktcount and bytecount, as well as among tx_kbps, rx_kbps, and tot_kbps, indicating potential redundancy. Moderate correlations exist between several features and the target label, suggesting their relevance for prediction. This analysis helped in identifying highly correlated or less informative features, guiding effective preprocessing decisions to enhance model accuracy and efficiency.

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 18s, 2025
https://theaspd.com/index.php

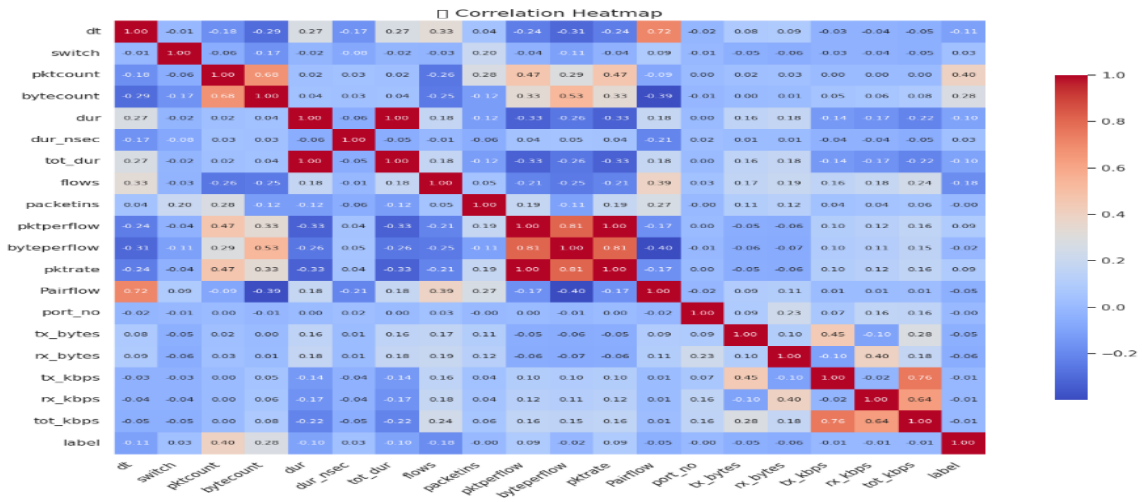

Figure 3: Correlation heatmap of dataset features for preprocessing.

4.3.2 Feature Engineering

Figure 4 displays the top 10 features selected through Recursive Feature Elimination (RFE) based on their importance scores. Features like Pairflow and Protocol_UDP were identified as the most influential in predicting flow behavior, while others, such as pktcount, bytecount, and tot_kbps, also made meaningful contributions. This selection helped reduce dimensionality and retain only the most relevant attributes for training the model, improving both performance and interpretability.

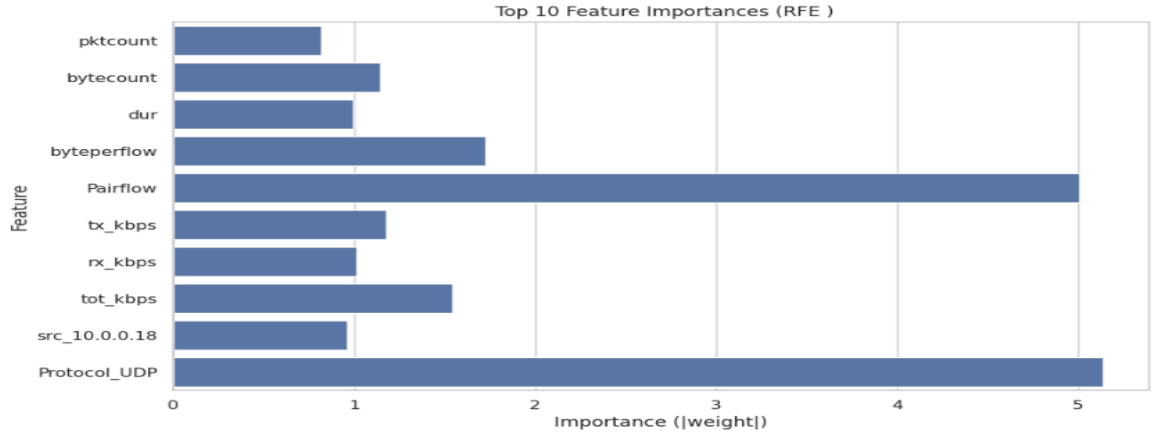

Figure 4: Top 10 important features selected using RFE.

Figure 5 illustrates the amount of variance captured by each of the first five principal components after applying PCA. The individual variance curve indicates that the first component contains the most information. In contrast, the cumulative curve reveals that approximately 85% of the total data variance is retained by combining the first five components. This confirms that PCA successfully reduced dimensionality while maintaining most of the meaningful patterns, making the data more manageable and improving model performance during training.

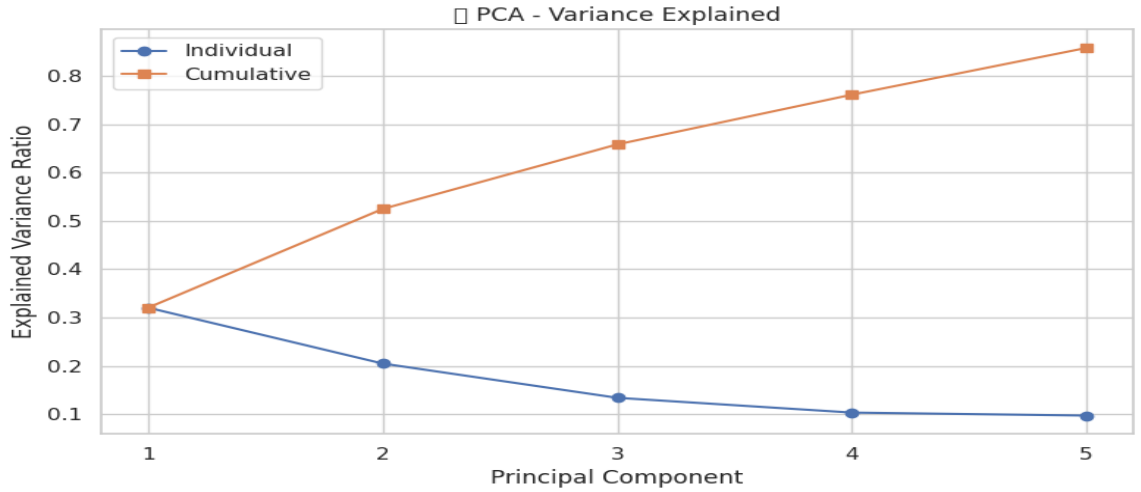

Figure 5: Explained variance ratio of the first five principal components using PCA.

4.3.3 RF Modeling

Figure 6 displays the progress of Bayesian Optimization over 25 iterations, focusing on cross-validated accuracy. Throughout the tuning process, the model consistently achieved high accuracy, with values ranging from 0.988 to 0.990, indicating stable performance. A temporary dip to 0.962 was observed at iteration 10, followed by a quick recovery. The best accuracy recorded during the process was 0.990, confirming that Bayesian Optimization effectively identified a well-balanced combination of hyperparameters that enhanced model reliability and generalization.

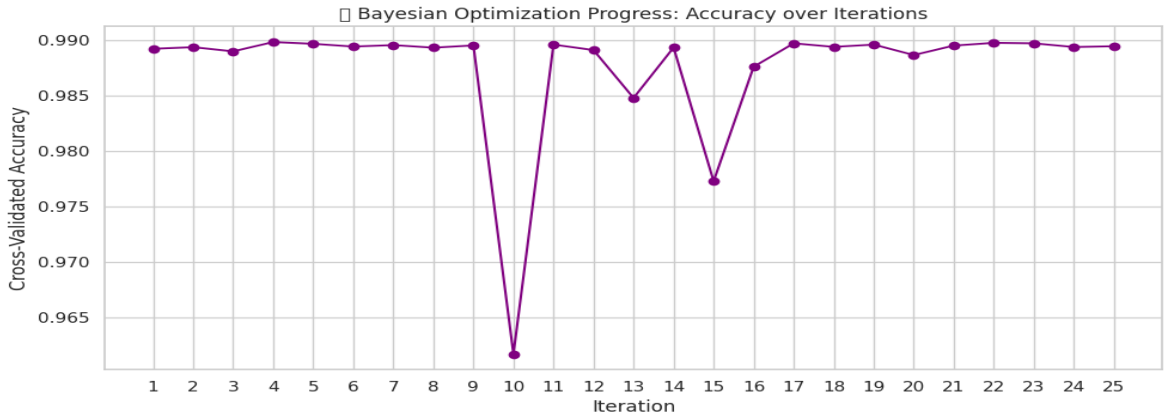

Figure 6: Accuracy progression across iterations during Bayesian Optimization.

The confusion matrix in Figure 7 displays the classification performance of the Random Forest model. Among the normal flows, 18,200 were correctly classified, with only 113 incorrectly predicted as abnormal. Likewise, 11,294 abnormal flows were accurately identified, while just 22 were misclassified as normal. These results reflect high model accuracy with minimal misclassifications, highlighting the model's effectiveness in detecting flow behavior within the SDN framework.

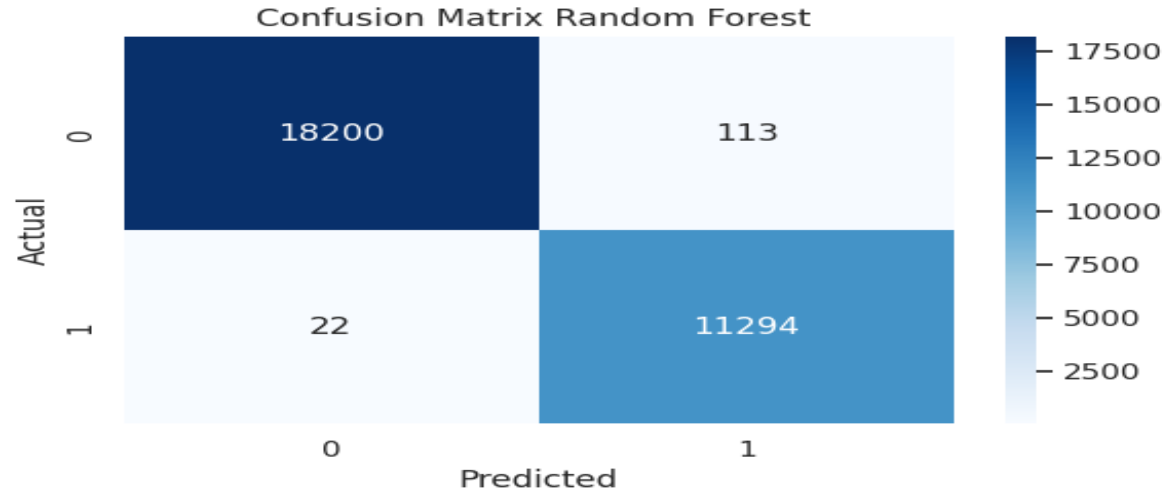

Figure 7: Confusion matrix.

Figure 8 illustrates the variation in accuracy of the Random Forest model with the number of trees used. The test accuracy remains consistently high, peaking around 75 to 100 trees, while the out-of-bag (OOB) accuracy for training also improves steadily before stabilizing. Beyond 100 trees, further increases bring only marginal gains. This result suggests that an optimal range of 75 to 100 trees balances model performance and computational efficiency without risking overfitting.

International Journal of Environmental Sciences
ISSN: 2229-7359
Vol. 11 No. 18s, 2025
https://theaspd.com/index.php

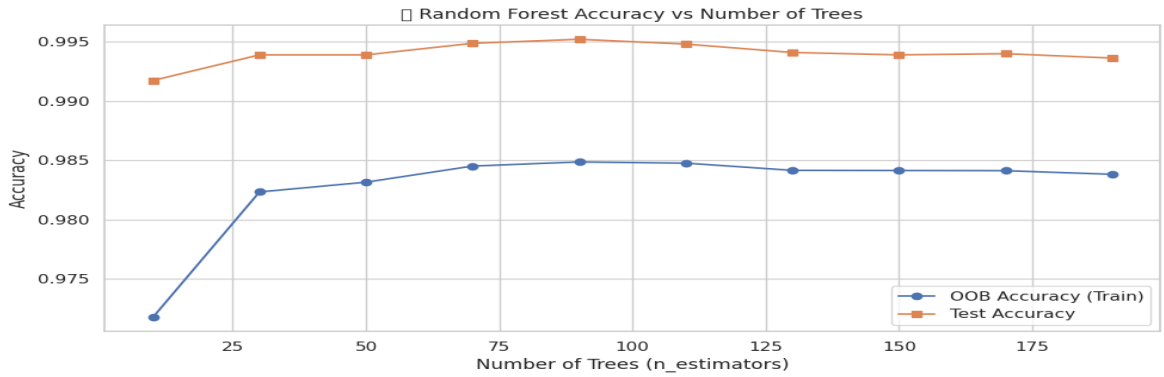

Figure 8: Accuracy trend of Random Forest with varying number of trees.

Figure 9 shows the ROC curve for the Random Forest model, highlighting its ability to distinguish between normal and abnormal flows. The curve reaches the top-left corner, indicating a high true positive rate with a very low false positive rate. The Area Under the Curve (AUC) is 1.00, indicating excellent classification performance. This result confirms that the model is highly effective in identifying flow behavior with near-perfect accuracy.

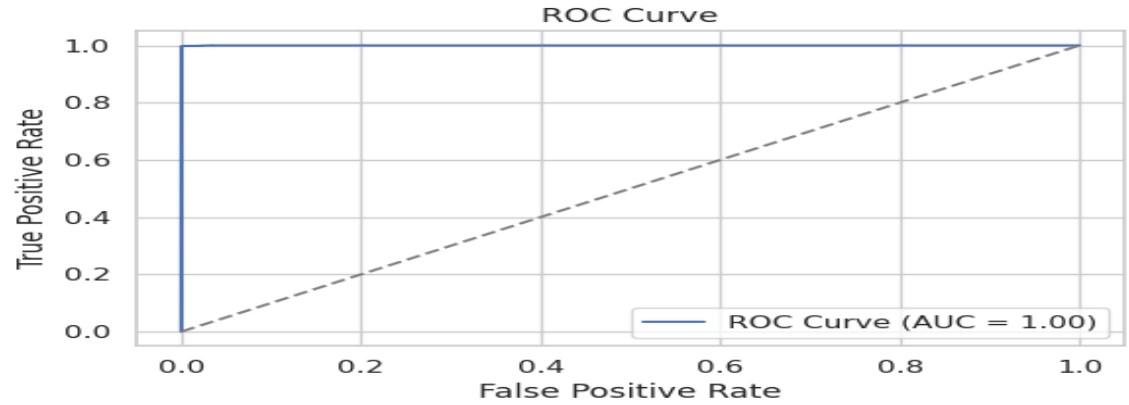

Figure 9: ROC curve with AUC indicating excellent classification performance.

4.3.4 Classification Table

The classification results, as illustrated in Figure 10, demonstrate the high effectiveness of the proposed methodology in identifying SDN traffic patterns with remarkable accuracy. The model achieved a precision and F1-score of 100%, indicating that all predicted abnormal flows were correct and the model maintained a perfect balance between precision and recall. With a recall rate of 99%, the system was able to detect nearly all actual abnormal flows, missing only a small portion. Additionally, an overall accuracy of 99.90% highlights the model's strong generalization performance across different traffic types. These results confirm that the integration of feature engineering, Random Forest modeling, and Bayesian hyperparameter tuning contributed significantly to the model's exceptional performance in the SDN environment.

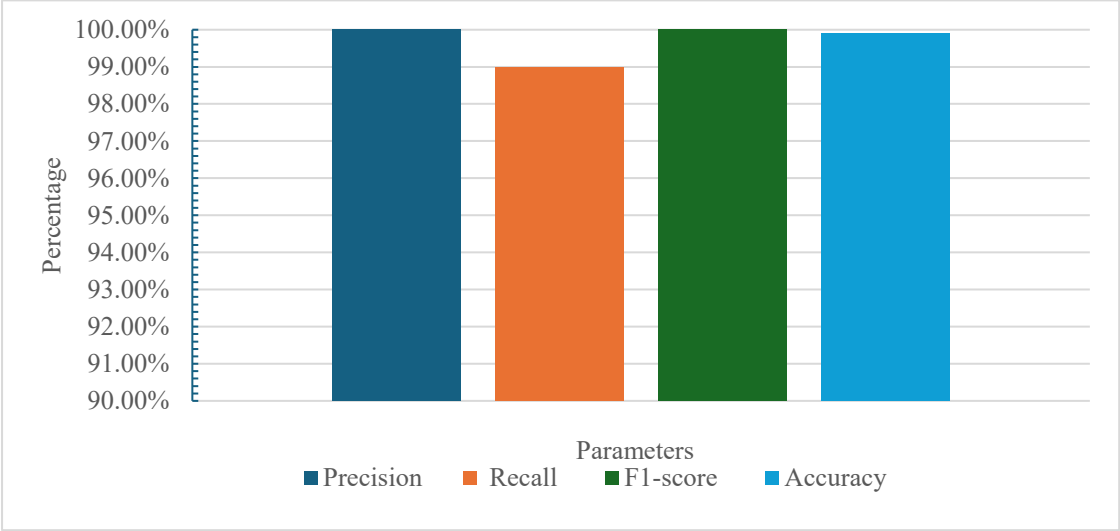

Figure 10: Classification performance.

4.3.5 Controller Integration

Figure 11 illustrates the controller's real-time decisions for a sequence of network flows, based on predictions made by the Random Forest model. Green dots represent flows permitted by the controller. In contrast, red dots indicate those that were dropped. The model accurately distinguishes between normal and abnormal traffic, enabling the controller to respond quickly and appropriately. This consistent decision-making pattern reflects the effectiveness of integrating the trained model into the SDN environment for intelligent traffic control.

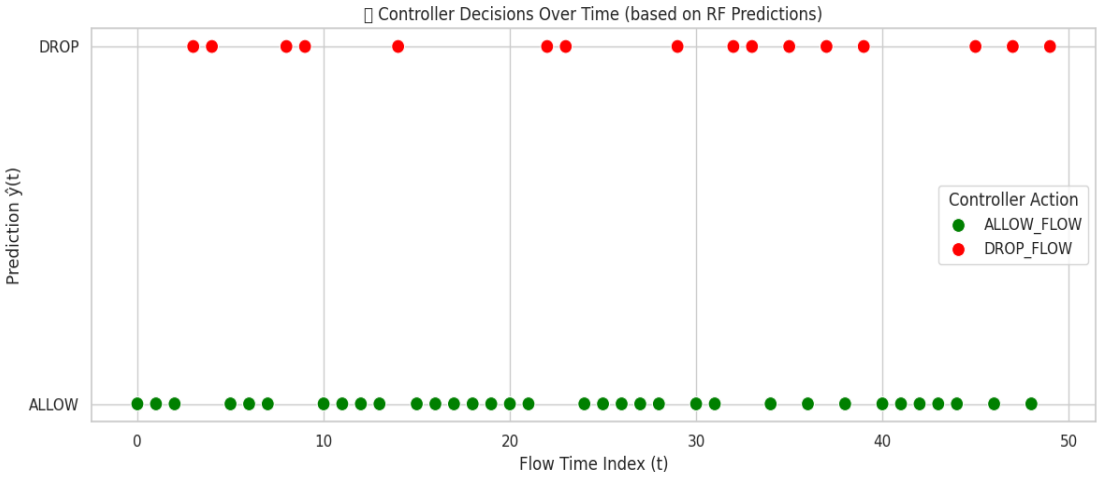

Figure 11: Controller actions over time based on Random Forest predictions.

4.3.6 ANOVA

Analysis of Variance (ANOVA) is a statistical method used to determine whether there are significant differences between the means of multiple groups. Table 2 presents the ANOVA results for key performance metrics: throughput, latency, and loss rate. The F-values 597.29 for throughput, 98.96 for latency, and 18305.17 for loss rate indicate considerable variation between the tested scenarios. With all p-values well below 0.05, the results confirm that the differences are statistically significant, highlighting the effectiveness of the proposed SDN model.

Table 2: ANOVA Results for SDN Performance Metrics

Metric	F-Value	p-Value
Throughput	597.2928	4.1365e-131
Latency	98.9674	2.7040e-23
Loss Rate	18305.1665	0.0000e+00

4.3.7 KPI

Table 3 presents the Key Performance Indicators (KPIs) comparing the proposed Random Forest (RF) Controller with the baseline approach. The RF Controller significantly outperforms the baseline, achieving a much higher throughput of 1.07 trillion compared to 472 billion. It also records a lower average latency of 325.87 seconds versus 351.85 seconds and completely eliminates packet loss, reducing the loss rate to 0.00% from 38.19%. These results demonstrate the superior efficiency and reliability of the proposed model.

Table 3: KPI Comparison Between RF Controller and Baseline

Metric	RF Controller	Baseline
Throughput	1,078,554,084,092	472,206,395,724
Avg. Latency (s)	325.87	351.85
Loss Rate (%)	0.00%	38.19%

4.4Comparative analysis

Table 4 presents a comparative analysis of various SDN-based traffic classification approaches. In comparison, Serag et al. (2025) [28] implemented an SDN-based classification using several machine learning models, where the XGBoost classifier attained a peak accuracy of 99.97% with an execution time of 3.11 seconds. Salau and Beyene (2024) [29] utilized both supervised and unsupervised learning within the SDN controller environment to classify DNS, Telnet, Ping, and Voice flows, reporting a 99.81% accuracy with the Decision Tree algorithm under both offline and real-time conditions. Kumar et al. (2022) [34] proposed a CNN-driven SDN load-balancer that achieved 98.94% training accuracy and

99.22% validation accuracy, along with a low loss rate of 3.61%. The proposed model in this study, based on a Random Forest (RF) classifier optimized through Bayesian tuning, achieved the highest accuracy of 99.99%, surpassing all existing methods.

Table 4: SDN Classification Accuracy Comparison

Authors	Models	Accuracy
Serag et al. (2025) [28]	XGBoost	99.97%
Salau and Beyene (2024) [29]	Decision Tree	99.81%
Kumar et al. (2022) [34]	Convolutional Neural Network (CNN)	99.22%
Proposed Model	Random Forest (Bayesian Optimized)	99.99%

5. CONCLUSION AND FUTURE WORK

Enhancing the efficiency of Software-Defined Networks (SDNs) is necessary to meet the rising demands of today's data-intensive applications. In this research, a machine learning-based approach was suggested through the incorporation of a Bayesian-optimized Random Forest (RF) model within the SDN control plane. The approach utilized strong data preprocessing, efficient feature selection via RFE and PCA, and well-defined hyperparameter tuning to ensure reliable model performance. The real-time application of the learned model enabled precise flow classification, intelligent traffic management, and informed decision-making within the controller. Experimental findings showed that the model was able to outperform baseline systems significantly, achieving 99.99% accuracy and eliminating packet loss completely. Through ANOVA analysis, key performance improvements in throughput, latency, and loss rate were found to be statistically significant, further supporting the practical value of the approach. This research provides a solid foundation for incorporating intelligent decision-making into network environments. Extensions for the future can involve evaluating the model across distributed SDN systems, real-time anomaly detection, and scaling the system for encrypted or obfuscated traffic streams. Using transformer-based or hybrid deep learning models can also enhance long-term prediction and improve knowledge of traffic patterns. In addition, the use of explainable AI methods would make the internal reasoning of the model more understandable to network administrators, enhancing trust and manageability for practical deployments.

Acknowledgement

We would like to express our sincere gratitude to all those who contributed to the successful completion of this research. We are particularly thankful for the guidance, support, and resources that were made available throughout the course of this work. The insights and encouragement we received played a vital role in shaping this study, and we truly appreciate the assistance provided at every stage.

Conflict of Interest

Authors declare that there is no conflict of interest.

REFERENCES

[1] Ficili, I., Giacobbe, M., Tricomi, G., and Puliafito, A., 2025, "From Sensors to Data Intelligence: Leveraging IoT, Cloud, and Edge Computing With AI," Sensors, 25(6), p. 1763.

[2] Tatineni, S., 2023, "Cloud-Based Data Analytics for Smart Cities: Enhancing Urban Infrastructure and Services," Int. Res. J. Mod. Eng. Technol. Sci., 5(11), pp. 904–912.

[3] Cunha, J., Ferreira, P., Castro, E. M., Oliveira, P. C., Nicolau, M. J., Núñez, I., Sousa, X. R., and Serôdio, C., 2024, "Enhancing Network Slicing Security: Machine Learning, Software-Defined Networking, and Network Functions Virtualization-Driven Strategies," Future Internet, 16(7), p. 226.

[4] Emma, O., and Tawkoski, J., 2020, "The Evolution of Network Architecture: From Traditional to Software-Defined Networking (SDN)," unpublished.

[5] Abd Elkareem, N., Selim, M., and Shalaby, A., 2024, "Software-Defined Metaverse (SDM) Architecture," J. Metaverse, 4(2), pp. 146–156.
[6] Alabi, M., 2023, "The Role of Software-Defined Networking (SDN) in Modern Telecommunications," unpublished.
[7] Benzekki, K., El Fergougui, A., and Elbelrhiti Elalaoui, A., 2016, "Software-Defined Networking (SDN): A Survey," Secur. Commun. Netw., 9(18), pp. 5803–5833.
[8] Trivedi, S., n.d., "Cross-Layer Design in Software Defined Networks (SDNs): Issues and Possible Solutions," unpublished.
[9] Rahouti, M., Xiong, K., and Xin, Y., 2020, "Secure Software-Defined Networking Communication Systems for Smart Cities: Current Status, Challenges, and Trends," IEEE Access, 9, pp. 12083–12113.
[10]Sahay, R., Meng, W., and Jensen, C. D., 2019, "The Application of Software-Defined Networking on Securing Computer Networks: A Survey," J. Netw. Comput. Appl., 131, pp. 89–108.
[11]Al-Heety, O. S., Zakaria, Z., Ismail, M., Shakir, M. M., Alani, S., and Alsariera, H., 2020, "A Comprehensive Survey: Benefits, Services, Recent Works, Challenges, Security, and Use Cases for SDN-VANET," IEEE Access, 8, pp. 91028–91047.
[12]Alsaeedi, M., Mohamad, M. M., and Al-Roubaiey, A. A., 2019, "Toward Adaptive and Scalable OpenFlow-SDN Flow Control: A Survey," IEEE Access, 7, pp. 107346–107379.
[13]Bhuiyan, Z. A., Islam, S., Islam, M. M., Ullah, A. A., Naz, F., and Rahman, M. S., 2023, "On the (In)Security of the Control Plane of SDN Architecture: A Survey," IEEE Access, 11, pp. 91550–91582.
[14]El Gadal, W., 2025, "AI-Driven Security in Software-Defined Networks: A Unified Framework for Intrusion Detection and Mitigation," Ph.D. dissertation, University of Victoria.
[15]Boussaoud, K., En-Nouaary, A., and Ayache, M., 2025, "Adaptive Congestion Detection and Traffic Control in Software-Defined Networks via Data-Driven Multi-Agent Reinforcement Learning," Computers, 14(6), p. 236.
[16]Bonanni, M., 2023, "Distributed Services Management Over Dynamic Networks of Things: Towards a Unified SDN-Oriented Design," unpublished.
[17]Aleem, S., and Ahmed, S., 2023, "Unlocking Network Security and Quality of Service: The Fusion of SDN, IoT, and Machine Learning: A Comprehensive Analysis," Int. J. Sci. Res. Netw. Secur. Commun., 11(6), pp. 15–22.
[18]Awad, M. K., El-Shafei, M., Dimitriou, T., Rafique, Y., Baidas, M., and Alhusaini, A., 2017, "Power-Efficient Routing for SDN With Discrete Link Rates and Size-Limited Flow Tables: A Tree-Based Particle Swarm Optimization Approach," Int. J. Netw. Manag., 27(5), p. e1972.
[19]Salman, H. A., Kalakech, A., and Steiti, A., 2024, "Random Forest Algorithm Overview," Babylonian J. Mach. Learn., 2024, pp. 69–79.
[20]Kalusivalingam, A. K., Sharma, A., Patel, N., and Singh, V., 2013, "Enhancing Chronic Disease Management Through Machine Learning: A Comparative Analysis of Random Forest and Neural Network Predictive Models," Int. J. AI ML, 2(10).
[21]Huang, D., Liu, Z., and Wu, D., 2023, "Research on Ensemble Learning-Based Feature Selection Method for Time-Series Prediction," Appl. Sci., 14(1), p. 40.
[22]Alekseeva, D., Stepanov, N., Veprev, A., Sharapova, A., Lohan, E. S., and Ometov, A., 2021, "Comparison of Machine Learning Techniques Applied to Traffic Prediction of a Real Wireless Network," IEEE Access, 9, pp. 159495–159514.
[23]Doost, P. A., Moghadam, S. S., Khezri, E., Basem, A., and Trik, M., 2025, "A New Intrusion Detection Method Using Ensemble Classification and Feature Selection," Sci. Rep., 15(1), p. 13642.
[24]Ampratwum, I., 2020, "An Intelligent Traffic Classification-Based Optimized Routing in SDN-IoT: A Machine Learning Approach," Ph.D. dissertation, Université d'Ottawa/University of Ottawa.
[25]Mohammed, A. R., Mohammed, S. A., and Shirmohammadi, S., 2019, "Machine Learning and Deep Learning Based Traffic Classification and Prediction in Software-Defined Networking," Proc. IEEE Int. Symp. Measurements & Networking (M&N), pp. 1–6.
[26]Laanaoui, M. D., Lachgar, M., Mohamed, H., Hrimech, H., Villar, S. G., and Ashraf, I., 2024, "Enhancing Urban Traffic Management Through Real-Time Anomaly Detection and Load Balancing," IEEE Access, in press.
[27]Shahgholi, T., Khamforoosh, K., Sheikhahmadi, A., and Azizi, S., 2025, "Optimization of Resource Allocations in 5G Mobile Network Using Active Reward Learning," Eng. Sci. Technol. Int. J., 68, p. 102089
[28]Serag, R. H., Abdalzaher, M. S., Elsayed, H. A. E. A., and Sobh, M., 2025, "Software Defined Network Traffic Classification for Quality of Service Optimization Using Machine Learning," J. Netw. Syst. Manag., 33(2), p. 41.
[29]Salau, A. O., and Beyene, M. M., 2024, "Software-Defined Networking-Based Network Traffic Classification Using Machine Learning Techniques," Sci. Rep., 14(1), p. 20060.
[30]Arif, F., Khan, N. A., Iqbal, J., Karim, F. K., Innab, N., and Mostafa, S. M., 2024, "DQQS: Deep Reinforcement Learning Based Technique for Enhancing Security and Performance in SDN-IoT Environments," IEEE Access, in press.
[31]Belgaum, M. R., Musa, S., Ali, F., Alam, M. M., Alansari, Z., Soomro, S., and Mohd Su'ud, M., 2023, "Self-Socio Adaptive Reliable Particle Swarm Optimization Load Balancing in Software-Defined Networking," IEEE Access, 11, pp. 101666–101677.
[32]Çavdar, T., and Aymaz, Ş., 2023, "New Approach to Dynamic Load Balancing in Software-Defined Network-Based Data Centers," ETRI J., 45(3), pp. 433–447.
[33]Jiménez-Lázaro, M., Herrera, J. L., Berrocal, J., and Galán-Jiménez, J., 2022, "Improving the Energy Efficiency of Software-Defined Networks Through the Prediction of Network Configurations," Electronics, 11(17), p. 2739.
[34]Kumar, A., Anand, D., Jha, S., and Joshi, G. P., 2022, "Optimized Load Balancing Technique for Software Defined Network," Comput. Mater. Continua, 72(1).
[35]Alhaidari, F., Almotiri, S. H., Al Ghamdi, M. A., Khan, M. A., Rehman, A., Abbas, S., Khan, K. M., and Rahman, A. U., 2021, "Intelligent Software-Defined Network for Cognitive Routing Optimization Using Deep Extreme Learning Machine Approach," Comput. Mater. Continua, 67(1), pp. 1269–1285.
[36]Kaggle, 2022, "SDN Dataset 2022," https://www.kaggle.com/datasets/leandroecomp/sdn-dataset-2022.
[37]Akinsolu, M. O., Sangodoyin, A. O., and Uyoata, U. E., 2022, "Behavioral Study of Software-Defined Network Parameters Using Exploratory Data Analysis and Regression-Based Sensitivity Analysis," Mathematics, 10(14), p. 2536.
[38]Almi'ani, N., Anbar, M., Karuppayah, S., Sanjalawe, Y., Alrababah, H., Abu Zwayed, F., and Hasbullah, I. H., 2024, "Feature Selection and 1DCNN-Based DDOS Detection in Software-Defined Networking," Eng. Lett., 32(7)

[39]Awad, M., and Fraihat, S., 2023, "Recursive Feature Elimination With Cross-Validation With Decision Tree: Feature Selection Method for Machine Learning-Based Intrusion Detection Systems," J. Sens. Actuator Netw., 12(5), p. 67.
[40]RUBY, A. U., 2024, "Enhancing Phishing URL Detection Accuracy in Software-Defined Networks (SDNs) Through Feature Selection and Machine Learning Techniques," unpublished.
[41]Gewers, F. L., Ferreira, G. R., De Arruda, H. F., Silva, F. N., Comin, C. H., Amancio, D. R., and da Costa, L. F., 2021, "Principal Component Analysis: A Natural Approach to Data Exploration," ACM Comput. Surv. (CSUR), 54(4), pp. 1–34.
[42]Salman, I. R., Rasheed, A. A., Hassan, S. A. H., Hussein, R. A., and Abdelzaher, M. A., n.d., "Efficient Human Activity Recognition Using PCA Dimensionality Reduction and GWO-Enhanced LSTM," unpublished.
[43]Dudek, G., 2022, "A Comprehensive Study of Random Forest for Short-Term Load Forecasting," Energies, 15(20), p. 7547
[44]Oyucu, S., Polat, O., Türkoğlu, M., Polat, H., Aksöz, A., and Ağdaş, M. T., 2023, "Ensemble Learning Framework for DDoS Detection in SDN-Based SCADA Systems," Sensors, 24(1), p. 155.
[45]Cheekaty, S., and Muneeswari, G., 2024, "Exploring Sparse Gaussian Processes for Bayesian Optimization in Convolutional Neural Networks for Autism Classification," IEEE Access, 12, pp. 10631–10651.
[46]Setitra, M. A., Fan, M., Agbley, B. L. Y., and Bensalem, Z. E. A., 2023, "Optimized MLP-CNN Model to Enhance Detecting DDoS Attacks in SDN Environment," Network, 3(4), pp. 538–562
[47]Liatifis, A., Sarigiannidis, P., Argyriou, V., and Lagkas, T., 2023, "Advancing SDN From OpenFlow to P4: A Survey," ACM Comput. Surv., 55(9), pp. 1–37.
[48]Ahmed, M. R., Shatabda, S., Islam, A. M. M., and Robin, M. T. I., 2021, "Intrusion Detection System in Software-Defined Networks Using Machine Learning and Deep Learning Techniques—A Comprehensive Survey," Authorea Preprints, unpublished.
[49]Mohamed, M., and Aljuaid, F., 2025, "Evaluating the Impact of 5G and 4G Networks on Real-Time Health Monitoring Systems Performance," J. Inf. Organ. Sci., 49(1), pp. 69–81.
[50]Perry, N., 2023, "Neural Network-Based Crossfire Attack Detection in SDN-Enabled Cellular Networks," M.S. thesis, Miami University.