

## **Proactive Packet Loss Management Using Random Forest and Real-Time Traffic Control**

**Avik Roy<sup>1</sup>, Sousnav Debnath<sup>2</sup>, Somsubhra Gupta<sup>3</sup>**

<sup>1</sup>Department of Computer Science and Engineering, Swami Vivekananda University, Barrackpore, West Bengal, India

<sup>2</sup>Department of Computer Science and Engineering, Swami Vivekananda University, Barrackpore, West Bengal, India

<sup>3</sup>Department of Computer Science and Engineering, Swami Vivekananda University, Barrackpore, West Bengal, India

### **Abstract**

Network performance is crucial these days where networking forms the backbone of modern communication, also supporting applications such as video streaming, real-time conferencing and gaming. Data transmission is essential for latency-sensitive services such as VoIP and video streaming, where any interruption can lead to a decline in user experience. Among these issues, packet loss stands as a significant issue which causes delays, jitters and reduces efficiency.

**Keywords :** Packet Loss Prediction, Machine Learning, AI, Random Forest, Network Optimization, Real- Time Network Monitoring, Load Balancing.

### **1. Introduction**

Networking has become a pivotal point for every application that requires high reliability and minimal latency. However, this reliability is challenging due to issues such as packet loss, where data packets fail to reach their destination leading to unsatisfactory service quality. This is crucial in applications such as online gaming, video streaming, and VoIP where even minimal loss can disrupt user experience.

This paper showcases a multi-step framework to address packet loss through AI/ML based prediction and mitigation strategies. The process begins with data collection and preprocessing, gathering information about network traffic patterns, latency and congestion. Next, a supervised learning model-Random Forest is used to predict packet loss based on historical data which offers an efficient solution. Then real-time traffic management techniques which includes load balancing and traffic shaping are used to minimize packet loss by optimizing network performance dynamically.

By combining predictive analysis with proactive traffic control, the proposed framework aims to enhance network resilience ensuring high-quality service in this digital age.

## 2. Background Study

Packet loss is a critical issue in networking which arises due to factors such as network congestion, faulty hardware or signal degradation which leads to significant disruptions in data transmission. Research highlights the role of machine learning models, particularly supervised learning algorithms in predicting and mitigating packet loss by analyzing network traffic patterns which offers a solution to enhance network performance



Fig. 1. network workflow

## 3. Framework Overview

### 3.1 The proposed solution is divided into three main parts:

#### 3.1.1. Data Collection and Preprocessing

Real-time data related to network traffic, packet loss, latency, and congestion is collected from network monitoring tools. This data is preprocessed to ensure that it is clean and usable for training the machine learning model.

### 3.1.2. AI/ML Model for Packet Loss Prediction

A Random Forest classifier is used to predict the chances of packet loss based on traffic and network conditions.

The model is trained from historical data to identify patterns which indicates packet loss events.

### 3.1.3. Traffic Management for Packet Loss Mitigation

When packet loss is predicted, the system will use load balancing to manage the network effectively and reduces congestion that leads to packet loss.

## 4. Data Collection and Preprocessing

### 4.1. Data Collection

Relevant data about network traffic will be collected in this step such as packet loss occurrences and other performance metrics. The following types of data will be collected:

- **Network Traffic Data:** This will include packet size, transmission rate, number of packets sent per unit time, bandwidth utilization, etc. This data will help to understand the current network load.
- **Packet Loss Data:** Historical records of packet loss incidents including timestamps, the extent of loss and network conditions at the time. This data will be used for training the machine learning model.
- **Latency and Jitter:** Measures of the delay in packet transmission and variability, both of which are important aspects of packet loss.
- **Environmental Data:** This will include status reports from network devices and connection types.

These data points will be gathered using tools like scapy.all or psutil and custom network monitoring APIs.

### 4.2. Data Preprocessing

Once the data is collected, it undergoes a preprocessing phase:

- **Cleaning:** In this step, we will remove all outliers, erroneous data points and missing values. This step will

ensure the integrity of the dataset.

- **Feature Engineering:** This step will create new features from the collected data that will be more predictive of packet loss. Example:

1. **Time Features:** Day of the week, time of day, etc., as traffic patterns vary with time.

2. **Traffic Features:** Average packet size, packets per second, and total throughput.

- **Normalization/Scaling:** Different features may have different units so normalization techniques like Min- Max scaling are applied to bring the data to a common scale which will ensure fairness during model training.

```
import pickle
from copy import deepcopy
from datetime import datetime
import random
from sklearn.preprocessing import MinMaxScaler

# Lists to store packet data
data = {'timestamp': [], 'packet_size': [], 'packets_per_second': [], 'total_throughput': [], 'packet_loss': []}

def packet_callback(packet):
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    packet_size = len(packet)
    net_io = psutil.net_io_counters()
    total_packets = net_io.packets_sent + net_io.packets_recv
    throughput = total_packets * packet_size

    # Append data to lists
    data['timestamp'].append(timestamp)
    data['packet_size'].append(packet_size)
    data['packets_per_second'].append(total_packets)
    data['total_throughput'].append(throughput)
    data['packet_loss'].append(0) # Simulated packet loss
    print(packet_summary())

# Sniff packets
sniffiface="eth0", promiscuous=True, count=500

# Display collected data
for i in range(len(data['timestamp'])):
    print("\nTimestamp: %s, Packet Size: %s, Packets per Second: %s, Total Throughput: %s, Packet Loss: %s" %
          (data['timestamp'][i], data['packet_size'][i], data['packets_per_second'][i], data['total_throughput'][i], data['packet_loss'][i]))

# Preprocess data
df = pd.DataFrame(data, dtype=object)
df = df[df['packet_size'] > 0]
df['timestamp'] = pd.to_datetime(df['timestamp'])
df['day_of_week'] = df['timestamp'].dt.dayofweek
df['hour_of_day'] = df['timestamp'].dt.hour
scaler = MinMaxScaler()
df[['packet_size', 'packets_per_second', 'total_throughput']] = scaler.fit_transform(
    df[['packet_size', 'packets_per_second', 'total_throughput']])

# Display preprocessed data
print("\nPreprocessed Data:")
print(df)
```

Fig. 2. Real-time network data manipulation

## 5. AI/ML Model for Packet Loss Prediction

### 5. 1. Model Selection: Random Forest

In this step, the Random Forest classifier will be used due to efficiency in handling datasets with complex relationships. Random Forest is an ensemble learning method that combines multiple decision trees to make predictions which offers advantages such as:

- **Robustness to Overfitting:** Random Forest can handle noisy data and prevent overfitting which will make it suitable for this problem.

- **Interpretability:** While Random Forest models are relatively easy to interpret compared to deep learning models, they still capture complex, non-linear relationships between features.

## 5.2. Model Training

The data will be split into a training set and a testing set. The training set will be used to train the Random model which is then validated on the test set to make sure that it will generalize well to unseen data. Hyperparameters like the number of trees, the depth of each tree and the number of features considered at each split will be optimized using grid search and cross-validation.

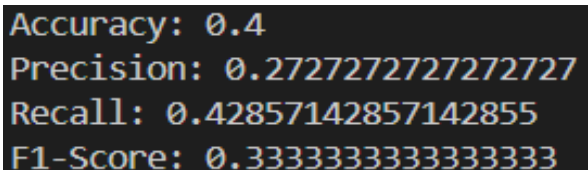
## 5.3. Model Evaluation

To evaluate the performance of the model, the following metrics will be used:

- **Accuracy:** It will include the percentage of correct predictions made by the model.

- **Precision, Recall, and F1-Score:** These metrics will be especially useful in estimating the performance of the model when predicting the event of packet loss.

A confusion matrix will be used to visualize the model's true positives, false positives, true negatives, and false negatives.



```
Accuracy: 0.4
Precision: 0.2727272727272727
Recall: 0.42857142857142855
F1-Score: 0.3333333333333333
```

Fig. 3. performance matrices

**6. Traffic Management for Packet Loss Mitigation** When the model will predict potential packet loss, the next step will be to take measures to mitigate the predicted loss.

**These measures will include:**

### **6.1. Load Balancing**

Load balancing is the process of distributing network traffic evenly across multiple paths or servers. We will use Weighted Round-Robin to direct the traffic to less congested paths which will prevent packet loss due to overloaded network components.

- **Dynamic Load Balancing:** If the AI model predicts congestion or packet loss in a certain path, traffic is dynamically rerouted to an alternative path to avoid the issue.

### **6.2. Steps for Implementing Load Balancing to Mitigate Packet Loss**

#### **6.2.1. Monitor Network Traffic**

The model will continuously track real-time traffic flows using network monitoring tools to detect high traffic volumes and potential bottlenecks.

#### **6.2.2. Identify Congestion Points**

The predictions made by the model or rule-based thresholds will be used to identify network paths or nodes experiencing high congestion that may lead to packet loss.

#### **6.2.3. Implement Load Balancing Algorithm**

We will deploy a load balancing Weighted Round- Robin:

- *Weighted Round-Robin:* It will distribute traffic based on the capacity or weight of each server or path.

#### **6.2.4. Reroute Traffic Dynamically**

Automatically the traffic will be redistributed from congested paths to alternate routes with sufficient capacity to handle additional load. Dynamic adjustments will be made based on current network conditions.

6.2.5. Integrate AI Prediction

We will use AI models to predict traffic spikes and preemptively adjust load balancing strategies to mitigate anticipated packet loss.

6.2.6. Monitor Post-Rerouting Performance

It will continuously evaluate the effectiveness of load balancing adjustments by monitoring metrics like latency and packet loss rate. This strategy will be refined based on observed outcomes.

By balancing the load across available paths, this approach will minimize congestion and reduce the chances of packet loss and ensures efficient use of network resources.

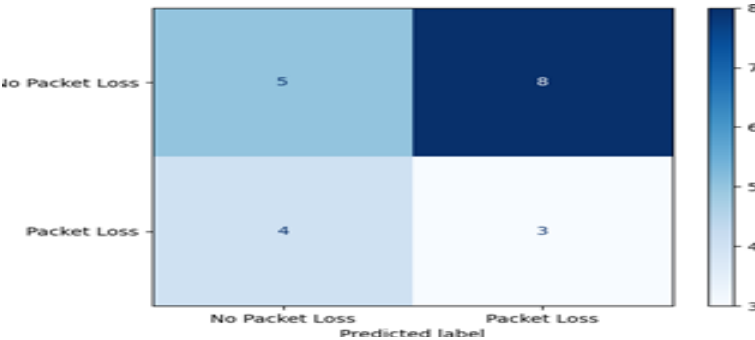


Fig. 4. predicted table

7. Real-Time Prediction and Deployment

7.1. Real-Time Monitoring and Integration

The trained model will be integrated into the network’s real-time monitoring system. As network traffic data is continuously fed into the system, the model will make predictions on packet loss events. If the chances of packet loss is high, the system will trigger preemptive actions such as adjusting load balancing parameters.

7.2. Alerts and Notifications

When packet loss is predicted, an alert system will be activated. Notifications will be sent to network administrators through email. This will allow network operators to make

manual adjustments or verify that the automated system is functioning as intended.

## **Conclusion**

The integration of AI/ML-based packet loss prediction and traffic management techniques represents a significant advancement in network performance optimization. By using Random Forest for packet loss prediction and implementing load balancing, networks can minimize packet loss which will ensure better performance and user experience especially for latency-sensitive applications.

The system provides network administrators with the ability to monitor network conditions in real time and make better decisions to enhance network reliability.

Future work will improve the model by incorporating additional data sources, such as application layer information and more advanced AI techniques to further improve the prediction accuracy and network optimization strategies.

## **References**

Congestion or No Congestion: Packet Loss Identification and Prediction Using Machine Learning by Inayat Ali, Seungwoo Hong, Taesik Cheung

Load Balancing Techniques for Efficient Traffic Management in Cloud Environment by Talasila Sasidhar, Vani Havisha, Sai Koushik, Mani Deep, V Krishna Reddy

A Machine Learning Approach for Packet Loss Prediction in Science Flows Anna Giannakoua , Dipankar Dwivedib and Sean Peiserta

Study of Packet Loss Prediction using Machine Learning by Dr. Kalpana Saha (Roy) , Tune Ghosh

Study of Traffic Based Load Balancing Algorithm in the SDN by Shalini Kumari and Sandeep Singh Kang

Packet Loss Prediction Using Artificial Intelligence Unified with Big Data Analytics, Internet of Things and Cloud Computing Technologies by Rishabh Chauhan and Sunil Kumar

Packet Loss Rate Prediction Using the Sparse Basis  
Prediction Model by A. Atiya, S. Yoo, K. Chong, Hyongsuk  
Kim

The Study On Load Balancing Strategies In Distributed  
Computing System by Md Firoj Ali

A Network Load Balancing Algorithm Design Based on  
Big Data by Lingfang Huang and Yi Shu

Load balancing techniques in cloud computing  
environment: A review by Dalia Abdulkareem Shafiq,

N.Z. Jhanjhi and Azween Abdullah