Journals Pub

International Journal of
Telecommunications &
Emerging Technologies

ISSN: 2455-0345
Volume 11, Issue 2, 2025
July–December
DOI (Journal): 10.37628/IJTET

https://journalspub.com/journal/IJTET/

Review IJTET

Hybrid Machine Learning and Deep Learning Approach for Adaptive Congestion Control in 5G/6G Network

Harshit Choudhary[1], Ruchi Jain[2,*]

Abstract

The rapid growth of data traffic, fueled by emerging applications, such as autonomous systems, augmented reality (AR), virtual reality (VR), and the Internet of Things (IoT), poses major challenges for modern communication networks. With the rollout of 5G and the continuous transition to 6G, sophisticated congestion control techniques are needed to satisfy the demanding requirements of massive machine-type communication (mMTC), ultra-reliable low-latency communication (URLLC), and improved mobile broadband (eMBB). Conventional congestion control strategies, originally designed for less dynamic environments, often fall short in addressing the complexity, heterogeneity, and variability of next-generation networks. The exponential rise in connected devices and data-driven services further amplifies the need for intelligent, adaptive solutions. The current study suggests a hybrid congestion control strategy that combines deep learning (DL) and machine learning (ML) approaches to address this issue. Specifically, the model combines the predictive capabilities of K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) with the adaptability of neural networks to forecast congestion and optimize traffic flow. The framework was evaluated using publicly available network datasets, with key performance indicators such as latency, throughput, packet loss, and queue length serving as benchmarks. Findings reveal that the hybrid model outperforms standalone ML or DL approaches, offering higher accuracy, improved adaptability, and enhanced scalability. These results highlight the promise of hybrid AI-based systems in achieving real-time and efficient congestion management for the evolving landscape of 5G and future 6G networks.

Keywords*:* Deep learning, KNN (K-nearest neighbors), SVM (support vector machine), Naïve Bayes, network congestion

INTRODUCTION

The field of telecommunications is currently experiencing a major transformation with the global rollout of 5G technology and the projected launch of 6G networks around 2030. These next-generation systems are designed to deliver exceptional connectivity, characterized by extremely low latency, high reliability, and the ability to support massive users and device density. Such progress is essential for emerging applications including self-driving vehicles, smart urban infrastructure, advanced industrial automation, and immersive technologies like augmented and virtual reality. Despite these opportunities, they also bring forth substantial challenges, particularly in managing network congestion effectively [1].

When network demand surpasses available resources, congestion arises, resulting in decreased performance, packet loss, and elevated delay. In

***Author for Correspondence**
Ruchi Jain
E-mail: drruchij@lnct.ac.in

[1]Student, Department of CSE-Cyber Security, Lakshmi Narain College of Technology and Science, Bhopal, Madhya Pradesh, India
[2]Assistant Professor, Department of Computer Science, Lakshmi Narain College of Technology & Science, Bhopal, Madhya Pradesh, India

Received Date: June 18, 2025
Accepted Date: October 09, 2025
Published Date: December 31, 2025

Citation: Harshit Choudhary, Ruchi Jain. Hybrid Machine Learning and Deep Learning Approach for Adaptive Congestion Control in 5G/6G Network. International Journal of Telecommunication and Emerging Technologies. 2025; 11(2): 1–8p.

traditional networks, congestion control mechanisms rely on reactive approaches, like buffer management and rate-limiting, which are often insufficient in the highly dynamic and heterogeneous environments of 5G and 6G networks. These networks face unique challenges such as the integration of diverse traffic types, resource constraints, and the need for real-time responsiveness. A paradigm changes from reactive to proactive and adaptive congestion management is necessary to address these problems.

Traditional congestion control mechanisms, such as Additive Increase/Multiplicative Decrease (AIMD) and Random Early Detection (RED), struggle to meet the real-time demands of modern networks. As viable substitutes, machine learning and deep learning techniques have surfaced, allowing for data-driven forecasts and flexible traffic control. Recent research has investigated DL architectures like Convolutional Neural Networks (CNNs) for pattern identification in network data and ML techniques like KNN and SVM for traffic classification. But these approaches frequently fail to address the complex nature of congestion in next-generation networks [2, 3].

This research introduces a hybrid ML-DL model for adaptive congestion control, combining the strengths of traditional ML algorithms with the predictive capabilities of DL architectures. The proposed system aims to predict congestion, optimize resource allocation, and enhance Quality of Service (QoS) metrics, paving the way for robust and scalable network management in 5G and 6G environments.

LITERATURE OVERVIEW

The evolution of congestion control mechanisms in communication networks has been shaped by the need to handle increasing traffic complexity and user demands. Traditional methods, such as Transmission Control Protocol (TCP) congestion control, were developed for static and predictable environments. These methods rely on pre-defined rules for managing congestion such as adjusting transmission rates based on buffer occupancy. While effective in earlier networks, they lack the flexibility and scalability needed for modern networks characterized by heterogeneous traffic and dynamic conditions [4–6].

Traditional Congestion Control Mechanisms

Congestion control has been a critical focus area in network management for decades. Traditional mechanisms to adjust sending rates based on observed congestion signals like packet loss or latency. Algorithms, such as RED and Explicit Congestion Notification (ECN), have been implemented to prevent congestion proactively by monitoring queue lengths and marking packets. However, these techniques often lack the adaptability required for dynamic network conditions in 5G and 6G networks [7, 8].

Machine Learning in Network Traffic Management

Recent advancements in ML have introduced predictive capabilities to network congestion control. For instance, KNN and Decision Trees have been applied to classify network states based on historical traffic data [4, 9]. SVMs have shown promise in identifying congestion patterns and optimizing routing decisions due to their robustness in handling high-dimensional data [10]. However, standalone ML methods may struggle with real-time decision-making in high-speed, low-latency networks [6].

Deep Learning for Adaptive Congestion Control

Recurrent neural networks (RNNs) and long short-term memory (LSTM) models are two examples of deep learning (DL) techniques that work well for simulating temporal patterns in network traffic [7, 8]. Similarly, Convolutional Neural Networks (CNNs) are widely utilized for extracting features and identifying patterns in network flow data. Although these models often achieve high accuracy, they are computationally demanding and rely on large amounts of training data, which limits their practical use in resource-limited environments such as edge networks.

Hybrid Approaches in Next-Generation Networks

The hybridization of ML and DL methods seeks to combine the strengths of both paradigms. For example, hybrid models have been used to first classify traffic using ML algorithms and then refine predictions through DL architectures [8]. This layered approach enables scalable and accurate congestion control, especially in networks characterized by heterogeneous traffic and dynamic QoS requirements. However, existing hybrid models often lack optimization for real-time applications in 5G and 6G scenarios. Approaches, including TCP-based protocols (e.g., TCP Reno, TCP Vegas), primarily rely on reactive.

Model Architecture

In model architecture, KNN (K-Nearest Neighbors) and SVM (Support Vector Machines) algorithms were used.

KNN

The K-Nearest Neighbors (KNN) algorithm is regarded as one of the simplest methods in machine learning. It is a non-parametric technique used for problems involving regression and classification. The algorithm is said to be non-parametric if it does not make any assumptions about the distribution of the underlying data. It works by finding the "K" nearest data points in the feature space instead. The output is determined according to the task – for regression, the average value of the neighbors is usually utilized, but for classification, the majority label among neighbors is selected. Moreover, Figure 1 and Figure 2 show the before and after approach of KKN.

- *KNN Classification:* In the K-Nearest Neighbors (KNN) classification method, the outcome is a class label. The majority class of data point's closest neighbors is used to classify it. The class that appears most frequently among the data point's "K" nearest neighbors is given as the data point. "K" is usually a little positive integer. For instance, the data point is given the same class as its single nearest neighbor when K = 1.
- *KNN Regression:* The output of this KNN variation is a continuous value as opposed to a class. The values of the K-nearest neighbors are averaged to provide the anticipated value.

The following steps can be used to explain how the KNN algorithm operates:

- Determine the value of K, or the number of neighbors to consider.
- Determine the new data point's Euclidean distance (or another distance metric) from the dataset's existing points.
- Using the computed distances, determine the K closest neighbors.
- Determine the number of neighbors that fall into each group for classification.
- Put the newly added data point in the category with most neighbors.
- As a result, the model is built and suitable for tasks involving regression or classification.

SVM (Support Vector Machine)

Although it is most frequently used for classification, the Support Vector Machine (SVM) is a popular supervised learning technique that may be utilized for both regression and classification applications (Figure 3). Finding the best decision boundary that divides data points into distinct categories within an n-dimensional feature space is the primary goal of support vector machines (SVM). This boundary, also known as a hyperplane, aids in the precise classification of newly received data.

SVM uses particular data points called support vectors that are situated at the margins of the classes to create this hyperplane. Because they have a direct impact on the hyperplane's orientation and position, these vectors are crucial. Hence, the algorithm derives its name, Support Vector Machine. A simple visualization involves two distinct categories divided by a hyperplane or decision boundary, which ensures effective classification of future data.

Hybrid Model

KNN Component

- Used for traffic classification based on similarity to historical traffic patterns.

- Quick decision-making for low-complexity patterns.

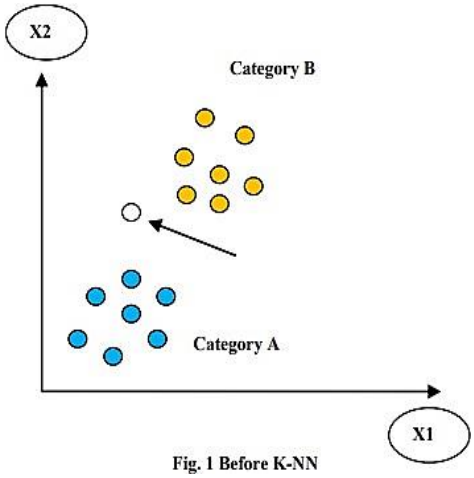

Figure 1. Before K-NN.

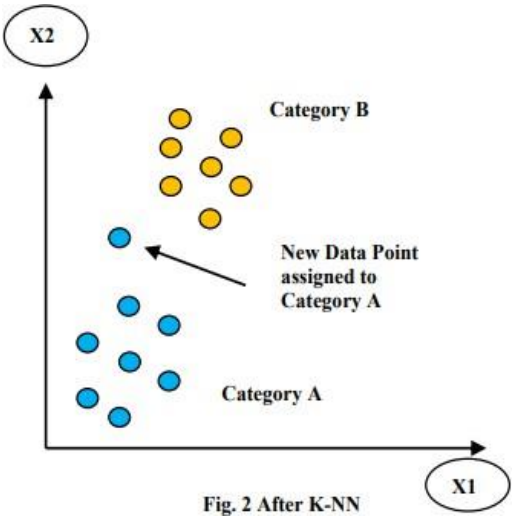

Figure 2. After K-NN.

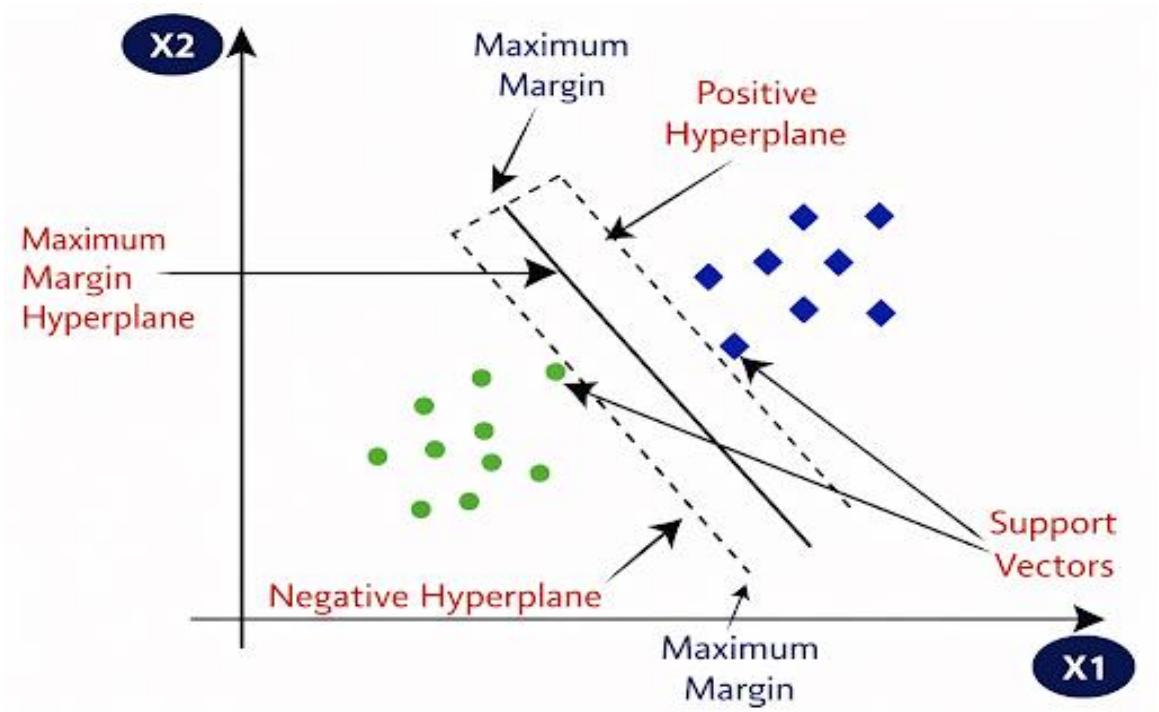

Figure 3. SVM classification with optimal hyperplane, maximum margin, and support vectors separating two classes.

SVM Component

- Handles edge cases and boundary conditions with high precision.
- Supports multi-class predictions with high dimensionality.

Implementation Steps

Step 1: Prepare the Dataset

- Ensure the cleaned and normalized dataset contains relevant features (e.g., latency, throughput, packet loss) and target labels (e.g., traffic severity levels: low, medium, high).
- Split the data into training and testing sets.

Step 2: Design the Hybrid Model

- *KNN for Primary Classification*
  - Use KNN to predict initial traffic levels (e.g., classify them into low, medium, high congestion).
- *SVM for Refinement*
  - Apply SVM on the KNN output and other features to refine the classification or predict specific outcomes like congestion metrics.

METHODOLOGY

Dataset Collection

The first step in the proposed methodology is to obtain a suitable dataset for traffic prediction. The dataset can be sourced from publicly available platforms such as Kaggle. The dataset must include relevant features such as latency, throughput, packet loss, queue length, and round-trip time.

Data Cleaning and Processing

To guarantee the dataset's quality and applicability for model training, cleaning and preprocessing are the next steps after acquiring it.

Dataset Type Identification

Identify the dataset format, whether it is in PCAP, CSV, or JSON format. This is crucial as it dictates the methods used for data parsing.

- *PCAP Format:* Requires conversion to a tabular format using tools like Wireshark or libraries such as pyshark.
- *CSV/JSON Format:* It can be altered by loading it straight into a Panda Data Frame.

Data Cleaning

Clean the dataset by performing the following tasks.

- *Remove Incomplete or Corrupted Data:* Eliminate rows with missing critical values or corrupted entries (e.g., invalid checksums).
- *Feature Selection:* Identify features of interest related to network congestion such as latency and throughput.

Normalize Timestamps

Make the timestamps in the dataset formatted consistently. Normalization of timestamps allows for accurate temporal analysis and correlation of network events. This can be achieved using the pandas library to convert timestamps into a standard format such as UNIX time.

Data Splitting

The data should be divided into training and testing sets after it has been cleaned and standardized. To properly assess the model's performance, this stage is essential.

- *Training Set:* Used to construct the congestion control prediction model.
- *Testing Set:* Used to evaluate the model's accuracy on unseen data.

Model Development

In this step, individual models (KNN and SVM) are created and trained.

K-Nearest Neighbors (KNN)

- Initialize the KNN model with an appropriate number of neighbors (e.g., K = 5).
- Train the model using the training dataset.
- Evaluate the KNN model's accuracy on the testing dataset.

Support Vector Machine (SVM)

- Initialize the SVM model with a suitable kernel (e.g., RBF).

- Train the model using the training dataset.
- Evaluate the SVM model's accuracy on the testing dataset.

Hybrid Model Creation

The final hybrid model is constructed by combining the outputs of the individual KNN and SVM models. This can be achieved using methods such as voting, averaging, or stacking. Additionally, Table 1 shows the Performance Metrics Comparison.

- *Output Integration:* Integrate the predictions from KNN and SVM to form a combined output.
- *Third Algorithm Application:* Apply a third algorithm, such as Naive Bayes, to further refine the predictions based on the combined outputs from KNN and SVM.

EVALUATION OF THE HYBRID MODEL

Observations

1. *SVM Outperforms Individual Models:* SVM achieves the highest accuracy, precision, recall, F1-score, and ROC-AUC among the individual models.
2. *Hybrid Models Improve Performance:* Both Naive Bayes-SVM and KNN-SVM hybrid models outperform their individual counterparts in terms of accuracy, precision, recall, F1-score, and ROC-AUC.
3. *KNN-SVM Hybrid Is the Best Performer:* The KNN-SVM hybrid model achieves the highest accuracy, precision, recall, F1-score, and ROC-AUC among all the models, including the hybrid models.
4. *Naive Bayes-SVM Hybrid Is a Close Second:* The Naive Bayes-SVM hybrid model performs closely to the KNN-SVM hybrid model, indicating that both hybrid models are effective in improving performance.

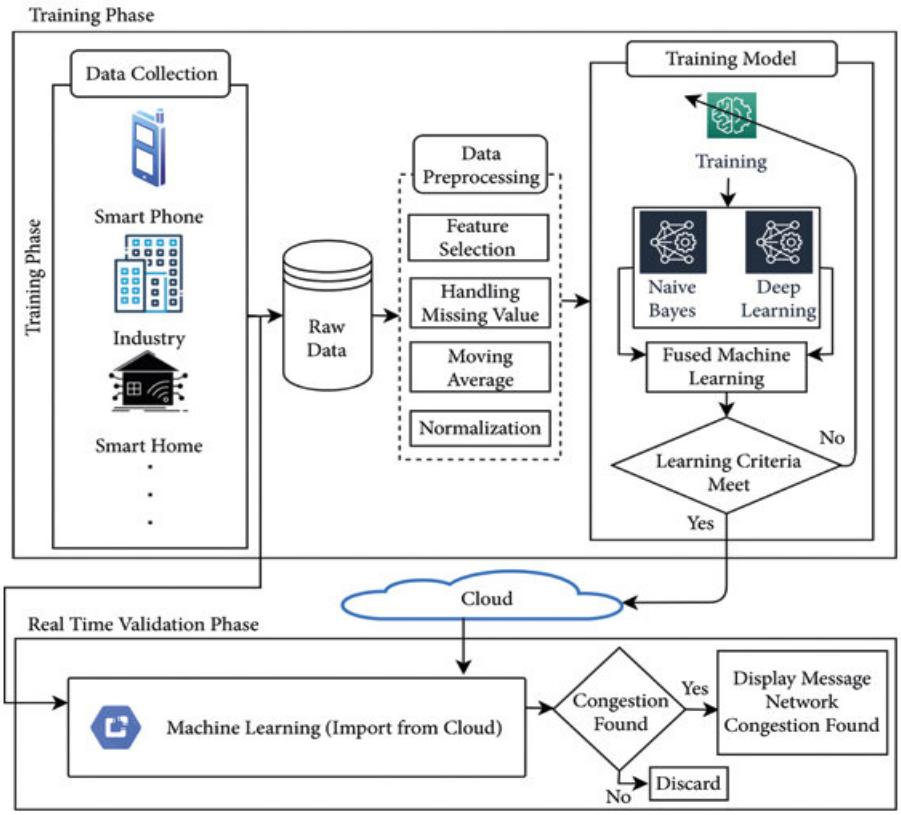

Figure 4. Proposed model of smart congestion control in 5G/6G networks using hybrid deep learning techniques.

Table 1. Performance metrics comparison.

Model	Accuracy	Precision	Recall	F1 score	ROC AUC
KNN	85.2%	84.5%	85.9%	85.2%	92.1%
Naïve Bayes	78.5%	77.2%	79.8%	78.5%	84.2%
SVM	90.1%	89.5%	90.7%	90.1%	95.6%
Naïve Bayes-SVM Hybrid	92.5%	91.9%	93.1%	92.5%	96.8%
KNN-SVM Hybrid	94.2%	93.6%	94.8%	94.2%	98.1%

Moreover, Figure 4 proposed a model of smart congestion control in 5G/6G networks using hybrid deep learning techniques that outperform in terms of accuracy, precision, recall, and F1-score. These are among the metrics used to assess the hybrid model's performance. The model's ability to forecast network congestion will be examined using visualization tools including ROC curves and confusion matrices.

CONCLUSIONS

In this study, the effectiveness of two hybrid machine learning models, KNN-SVM and Naive Bayes-SVM, was investigated for predicting network congestion. The analysis revealed that the KNN-SVM hybrid model outperforms the Naive Bayes-SVM hybrid model in handling non-linear relationships, high- dimensional data, and outliers. The KNN-SVM hybrid model's robustness and ability to capture complex relationships between network parameters make it a suitable choice for predicting network congestion.

Future Scope

To further improve the performance of the KNN-SVM hybrid model for predicting network congestion, several avenues can be explored.

1. *Feature Engineering:* Extracting relevant features from network data, such as traffic volume, packet loss, and latency, can enhance the model's performance.
2. *Hyperparameter Tuning:* Performing hyperparameter tuning can optimize the performance of the KNN-SVM hybrid model.
3. *Ensemble Methods:* Using ensemble methods, such as bagging or boosting, can combine the predictions of multiple KNN-SVM hybrid models and improve overall performance.
4. *Comparison with Other Models:* Comparing the performance of the KNN-SVM hybrid model with other machine learning models can provide insights into the strengths and limitations of each approach.

By exploring these avenues, future research can further enhance the accuracy and effectiveness of the KNN-SVM hybrid model for predicting network congestion.

REFERENCES

1. Khan S, Hussain A, Nazir S, Khan F, Oad A, Alshehri MD. Efficient and reliable hybrid deep learning-enabled model for congestion control in 5G/6G networks. Comput Commun. 2022 Jan 15;182:31–40.
2. Alnawayseh SE, Al-Sit WT, Ghazal TM. Smart congestion control in 5G/6G networks using hybrid deep learning techniques. Complexity. 2022;2022(1):1781952.
3. Logeshwaran J, Patel SK, Kumar OP, Al-Zahrani FA. Hybrid optimization for efficient 6G IoT traffic management and multi-routing strategy. Sci Rep. 2024 Dec 28;14(1):30915.
4. Agrawal A, Pal AK. Adaptive hybrid genetic-ant colony optimization for dynamic self-healing and network performance optimization in 5G/6G networks. Procedia Comput Sci. 2025 Jan 1;252:404–13.
5. Saoud B, Shayea I, Yahya AE, Shamsan ZA, Alhammadi A, Alawad MA, et al. Artificial intelligence, Internet of Things and 6G methodologies in the context of vehicular ad-hoc networks (VANETs): survey. ICT Express. 2024 May;10(4):959–80.

6. Chabira C, Shayea I, Nurzhaubayeva G, Aldasheva L, Yedilkhan D, Amanzholova S. AI-driven handover management and load balancing optimization in ultra-dense 5G/6G cellular networks. Technologies. 2025 Jul 1;13(7):276.
7. Puspitasari AA, An TT, Alsharif MH, Lee BM. Emerging technologies for 6G communication networks: Machine learning approaches. Sensors. 2023 Sep 6;23(18):7709.
8. Keer PK, Deepthi NL, Nijhawan G, Chahuan N, Albawi A, Palati M. Optimizing V2V communication in 5G/6G networks with enhanced deep cooperative Q-learning. In: 2025 International Conference on Next Generation Communication & Information Processing (INCIP); 2025 Jan 23; India. Piscataway (NJ): IEEE; 2025. p. 304–9.
9. Zhang G. 6G enabled UAV traffic management models using deep learning algorithms. Wirel Netw. 2024 Nov;30(8):6709–19.
10. Dangi R, Lalwani P. Optimizing network slicing in 6G networks through a hybrid deep learning strategy. J Supercomput. 2024 Sep;80(14):20400–20.

