International Journal of Artificial Intelligence,
ISSN 0974-0635; Int. J. Artif. Intell.
2025, Volume 23, Issue Number 2

Delay-driven Maximum Network Utilization by Machine Learning Model for Congestion Control

Keerti Mishra[1] and **Nitin Jain**[2]

[1,2] *Dept. of Electronics and Communication Engineering, School of Engineering, BBD University, Lucknow, INDIA*
Email : mishrakeerti2609@gmail.com[1], hod.ece@bbdu.ac.in[2]

ABSTRACT

The high speed growth of content-centric applications has drawn focus on efficient multi-source and multipath data transmission in Content-Centric Networking (CCN). The traditional approach for congestion control developed for stable, host-based architectures face several challenges due to CCN's dynamic, cache-driven environment. It leads to inaccurate congestion estimation, bandwidth underutilization, and unstable fairness. To address these challenges, this article has proposed a Delay-Driven Congestion Control Protocol (DDCCP) that utilizes delay and other network parameters as input indicators for machine-learning-based decision models for precise congestion level estimation. The proposed work maintains autonomous control, enabling fair and effective bandwidth distribution without requirement of router-specific link-layer information. Five ML models—Decision Tree, k-NN, Naive Bayes, Fuzzy Logic, and SVM—are evaluated to guide the delay-based congestion adjustments, with SVM achieving the highest accuracy and lowest delay. Simulation results demonstrate that DDCCP significantly improves bandwidth utilization, reduces packet loss, accelerates fairness convergence, and adapts efficiently to caching effects in multipath CCN environments. The proposed framework offers a scalable, intelligent, and high-performance alternative to current CCN congestion control strategies.

Keywords*:* Content-Centric Networking (CCN), Delay-Driven Protocol, Smart Congestion Control, Multipath Transmission.

Mathematics Subject Classification: 94A40

Computing Classification System: Networks~Network algorithms~Control path algorithms~Traffic engineering algorithms

1. INTRODUCTION

Content-driven applications are the main drivers of modern Internet activities. The introduction of a networking framework aims to enable more efficient data delivery (Zhang et al., 2014); (Bobrov et al. 2024). It utilizes URIs for forwarding information according to Interest and Data packets, in contrast to conventional models that rely only on host addresses. This change naturally supports a multipath and multisource transmission approach by allowing routing decisions to use URL-based identifiers and content replication through in-network caching (Mehta, 2021). However, the absence of address-based forwarding can lead to significant complications. Customers are unaware of the paths along which the data are forwarded. Furthermore, owing to different caching strategies, the fragmentation of cached data along the path is unstable. Existing multipath congestion control protocols such as

MPTCP (Wischik et al., 2011) and CMT-SCTP (Zheng et al, 2021) assume fixed endpoints and stable routing paths, making them unsuitable for decentralized and cache-driven CCN architectures. Current CCN congestion management approaches generally fall into two categories: hop-by-hop adaptive forwarding and receiver-driven rate control (Chen et al., 2016).In these designs, routers—not users—handle adaptive forwarding, leaving consumers with limited visibility and forcing them into coupled congestion control, where a congested path forces rate reduction across all paths, even when others remain underutilized. Congestion indicators used in various work are RTT-based estimation (Mahdian et al., 2016) or pending Interest counts (Bai et al., 2024). These indicators often observed to be imprecise and unable to account for cache-induced fluctuation in delay and throughput. Path-switching techniques proposed (Ye et al., 2018) offer greater user control over forwarding decisions but still requires optimization framework and struggle to leverage cache dynamics effectively. Similar limitations are observed in MIRCC ((Mahdian et al., 2016), which depends on router-side rate estimation and is difficult to scale. These constraints—combined with the impracticality of requiring link-layer visibility in overlay or wireless networks (Zhang et al., 2014); (Bai et al., 2022); (Chen et al., 2022) —highlight the need for a more adaptive and intelligent congestion control mechanism. Motivated by these challenges, this research introduces a DDCCP that integrates the machine-learning-based congestion estimation with, delay-oriented transport mechanism. The proposed framework provides congestion management and uses other parameters such as RTT, buffer free space, and partial network availability to infer congestion levels accurately. The ML models include Decision Tree, k-NN, Naive Bayes, Fuzzy Logic, and SVM—the proposed work enhances responsiveness, reduces dependence on router-specific measurements, and achieves faster convergence toward fair and efficient bandwidth utilization. Design of an appropriate adaptive forwarding strategy, in which routers dynamically divide traffic across all the routes to manage bottleneck loads, is the traditional method of increasing bandwidth utilization. Time-varying capacity limitations driven by cache hit behavior is highly faced under such optimum traffic distribution, which is considered as a multi-commodity flow issue (Bai et al., 2024). This model is computationally costly if used (Khanal, 2024). Because of the difficulties in measuring congestion rate, exact forwarding of data is still challenging, even with reduced heuristics. Some techniques proposed in past depend on imprecise estimations, such as the number of outstanding interests (OMCC-RF) (Bai et al., 2024), which lead to low utilization of bandwidth (Ye et al. 2018); (Ould et al., 2024); (Schneider et al., 2016). On the other hand, in order to achieve accuracy, routers must identify and keep track of bottleneck congestion and only route traffic via uncongested pathways, which adds a substantial computational and monitoring cost. Path switching (Moiseenko & Oran, 2017) gives users the ability to control data delivery on their own, offering another method of resolving bandwidth efficiency issues in networking. To improve network performance, the Path-specified Transport Protocol (PTP) (Ye et al., 2018) was established. It combines path switching with a congestion management mechanism similar to TCP. Nevertheless, PTP does not have a formal mathematical model to depict system equilibrium or network utility. In this research, a delay-driven congestion management technique is proposed and the optimal traffic distribution issue is formulated as a global optimization model. By preserving autonomous control over traffic on each route, the suggested framework preserves the fundamental

architecture of PTP. Furthermore it follows smart machine learning decision models by examining how network caching affects congestion and improving utilization of link.

The key contribution of this work is:

(1) Employed machine learning for maximum network utilization in data transmission for Content-Centric Networking (CCN) environment.

(2) Presented a high-performance Delay-driven Congestion Control specified by network parameters.

(3) Evaluated performance results that show faster bandwidth and fairness convergence.

The layout of this paper organized as follows: An overview of relevant work is given in Section II, and the key terms and concepts are defined in Section III. Delay-driven transmission using a network usage framework is covered in Section IV. In order to solve this problem, it presents a delay-aware method. Section V wraps up the study and explains the delay-based congestion management protocol design.

1.1. Research gap

Existing multipath congestion control schemes in Content-Centric Networking (CCN) environment protocols either rely on imprecise indicators like RTT and pending Interests or require link-layer visibility, making them impractical in dynamic overlay environments. The traditional protocols like MPTCP and SCTP assume stable endpoints; while path-switching models lack formal optimization and fail to effectively utilize caching status .Limited research exist on combining delay-driven indicators with machine learning for achieving robust congestion estimation, fairness, and bandwidth efficiency.

1.2. Problem statement

Content-centric traffic in multi-source multipath networks application is suffering from underutilization of bandwidth and unstable fairness due to inaccurate estimation of caching, variable delays, and decentralized path selection. Existing congestion control protocols either consider all the routes under monitoring and management and lead to global degradation performance, or depend on costly and inaccurate congestion measurement methods.

.2. RELATED WORK

Adaptive forwarding and congestion regulation are the two commonly used methods of traditional traffic control. In recent developments, a third developing category has evolved based on path switching.

2.1. Regulation of congestion

Congestion control is performed either by receiver-driven control or hop-by-hop selection based on user choice (interest). The receiver-driven control modifies the content request rate in reaction to congestion and sees all routes as a single entity. It observes the signals that include Congestion Notification, packet loss, and delay. Timeout methods were used in early work (Gao et al. 2024) to identify packet losses; however, estimations are incorrect (Zhang et al.; 2014) because of variable

latency. Another method that uses Round-Trip Time (RTT) [Bai et al., 2014, (Mahdian et al., 2016)], in which data packets are labeled with routing information, allows the user to identify and react to congestion based on path-based functions.

Control of the forwarding rate of packets is used in hop-by-hop forwarding to reduce data packet congestion. This is implemented by methods like HoBHIS (Bai et al., 2022); (Rozhnova and Fdida, 2014), which restrict interest transmission rates by equal allocation of bandwidth across different flows. For example, CHoPCoP (Zhang et al., 2014) controls this rate by examining the quantity of pending packets and connection-layer delays. The transmission rate is modified by following a Dynamic Interest Limiting (Chen et al., 2022) scheme on the basis of received Negative Acknowledgments (NACKs). HIS (Zhao et al., 2025) uses bandwidth information for the prior computation of the ideal interest forwarding rate. Despite satisfactory level of efficacy, the majority of techniques is dependent on access of link-layer parameters like channel capacity and queue size. If transmission is deployed on an overlay network, comprehensive link-layer information is not accessible; hence, this dependence becomes impractical.

2.2. Adaptive forwarding

By allowing users to select a network channel as a single, cohesive system, adaptive forwarding facilitates efficient transmission. In a scheme, Ant colony optimization is used for probability-based adaptive forwarding (Askar et al., 2023), which chooses a forwarding decision that depends upon round-trip time (RTT). Similar to this, On-demand Multi-Path Interest Forwarding (Udugama et al., 2014) uses RTT in conjunction with a round-robin strategy to distribute traffic among discontinuous pathways. Stochastic Adaptive Forwarding, a method that follows a fluid distribution network to accomplish traffic allocation, was introduced by Daniel et al. (Posch et al., 2016) improve resilience with inadequate routing information. The Request Forwarding Algorithm (RFA) was proposed (Bai et al., 2024), which uses the number of pending interests per interface as an indicator of congestion for handling problems associated with multi-path flow of data.

PCON (Schneider et al., 2016) improves overall multipath transmission efficiency by rerouting the high traffic links toward less crowded links by using the Explicit Congestion Notification (ECN) algorithm. With the exception of PCON, all data forwarding schemes generally faces issue of low bandwidth utilization because they depends on imprecise indicators like round-trip time (RTT) and the number of packets pending for transmission, which are not a reliable for measurement of actual level of congestion. According to experimental results it is observed that such ECN-based congestion balancing strategy is unstable and have converging issues. "Bottleneck oscillation" is the term used to characterize this behavior. Because the consumer drives congestion control in data networks, bandwidth needs naturally change over time, shifting bottlenecks and impacting the equilibrium condition of the network. Furthermore, ECN sometime degrades the demand swings and causes fluctuation in performance since it offers delayed feedback, and consequently reduces bandwidth efficiency.

Table 1: Literature Review Summary

	A. Regulation of Congestion			
Ref. No.	**Author (Year)**	**Approach**	**Pros**	**Cons**
[11]	Gao et al. (2024)	Fair flow congestion control for CCN	Promotes fairness and balanced bandwidth allocation	Fairness may reduce total throughput; requires parameter tuning
[12]	Zhang et al. (2014)	Explicit Congestion Control for CCN using RTT	Provides explicit congestion feedback	Adds signaling overhead, higher complexity
[14]	Bai et al. (2022)	DSCCP: Differentiated service-based congestion control in ICN	Supports service prioritization; enhances QoS	Low-priority traffic may suffer; higher computational cost
[16]	Chen et al. (2022)	ALBLP: Adaptive Load-Balancing using Link-State Prediction in SDN	Predictive algorithm reduces congestion and improves flow	Requires accurate prediction models; increase controller overhead
[17]	Zhao et al. (2025)	Hop-by-hop multi-topology traffic engineering for inter-datacenter WANs	Enhances efficiency and resilience; traffic optimization	High complexity in route computation; depends on accurate topology data
	B. Adaptive Forwarding			
[18]	Askar et al. (2023)	Performance evaluation of efficient Interest forwarding mechanisms in NDN over IEEE 802.15.4 networks	Improves forwarding efficiency in low-power, lossy networks	Limited to IEEE 802.15.4 environments (mainly IoT)
[19]	Udugama et al. (2014)	On-demand multi-path Interest forwarding	Combines RTT + round-robin, better distribution	High overhead, RTT dependency
[20]	Posch et al. (2016)	Stochastic Adaptive Forwarding (SAF)	Robust under limited routing info	Randomized forwarding may lower efficiency
[6]	Bai et al. (2024)	QSCCP: QoS-aware congestion control for ICN	Ensures QoS differentiation and congestion control	Higher complexity and overhead in implementation
[9]	Schneider et al. (2016)	PCON with ECN-based congestion balancing	Efficient traffic redirection to idle links	Prone to oscillations, unstable convergence
	C. Path Switching			
[13]	Mahdian et al. (2016)	MIRCC: Multipath-aware ICN rate-based congestion control	Enables per-flow rate control, path switching	Requires router-side rate estimation, scalability issues
[5]	Ye et al. (2018)	Path-specified Transport Protocol (PTP)	User-controlled path switching, TCP fairness	No mathematical model, limited cache utilization

2.3 path switching

High-speed forwarding is supported via path switching (Moiseenko and Oran, 2017), which gives users the ability to specifically select the transmission path. The consumer inserts a path label into the packet, which is then routed via the longest prefix match (LNPM) and the appropriate interface.

Path switching helps to improves scalability of system by elimination of complex adaptive forwarding method at routers. For enhancing the data transmission a mechanism suggested for congestion control called MIRCC (Mahdian et al., 2016) inspired by the Rate Control Protocol (RCP). In this method, customers can regulate traffic on a per-flow on the basis of transmission rate. The transmission rate estimate is forwarded by the receiver end. Limitations of this approach include: (1) scalability issues, as the router needs to calculate flow rates; (2) reliability on measurements like queue length, which might not be available; and (3) incompatibility with in-network caching.

Motivated through path switching concepts, this paper presents an algorithm that improves the loss-based MPTCP based strategy to control congestion, namely the Linked Increases algorithm (Wischik et al. 2011), to facilitate in-network caching and multipath transmission without requiring access to lower-layer link status data. According to the result findings, the proposed approach is capable of maintaining flexible forwarding better than current techniques and also helps in improving bandwidth usage. Research Background: Content-centric and information-centric networking paradigms shift the focus from host-based addressing to content-based delivery, naturally encouraging multipath and multi-source transmission. Traditional congestion management strategies—receiver-driven control, hop-by-hop shaping, and adaptive forwarding—have shown limitations in accuracy, scalability, and efficiency. Recent advances in machine learning provide adaptive decision-making capabilities that integrate with delay-driven protocols to overcome long-standing challenges in multipath congestion control.

3. METHODOLOGY

In order to address the problem of bandwidth underutilization, this study applies a unique notion of virtual parallelism in TCP to maintain the transmission of multiple parallel data streams. This method alters TCP Reno's Additive Increase Multiplicative Decrease (AIMD) mechanism. A 100 Mbps bottleneck link is used for the experimental assessment, which has an RTT of 80 ms and a packet loss rate of 0.02%. It shows that bandwidth that is underutilized in traditional TCP Reno can be efficiently employed by this technique. Nevertheless, the throughput of concurrent TCP Reno flows tends to decrease as the number of parallel streams (N) rises, suggesting synchronization problems and decreased fairness. The TCP-FIT (Boudi and Loudini, 2024) technique is presented as a dynamic adjustment mechanism for the value of N in order to solve this fairness problem. Even with this improvement, the correction is frequently insufficiently precise and not soon enough. This limitation drives the creation of a different approach, a TCP congestion control method based on machine learning. The round-trip time (RTT) of each ACK is utilized similar to the TCP Indigo protocol to determine the degree of network congestion at that moment. A decision-making framework employing machine learning algorithms applies established rules to dynamically manage congestion control behavior based on each incoming ACK.

Table 2: Rules for proposed machine learning decision model based cl estimation under TCP

RTT	BFSS	PNA	*CL*	RTT	BFSS	PNA	CL	RTT	BFSS	PNA	CL
Min	Min	Min	Max	Mid	Min	Min	Max	Max	Min	Min	Max
Min	Min	Mid	Mid	Mid	Min	Mid	Mid	Max	Min	Mid	Max
Min	Min	Max	Min	Mid	Min	Max	Mid	Max	Min	Max	Mid
Min	Mid	Min	Mid	Mid	Mid	Min	Mid	Max	Mid	Min	Mid
Min	Mid	Mid	Mid	Mid	Mid	Mid	Mid	Max	Mid	Mid	Mid
Min	Mid	Max	Mid	Mid	Mid	Max	Mid	Max	Mid	Max	Mid
Min	Max	Min	Min	Mid	Max	Min	Mid	Max	Max	Min	Mid
Min	Max	Mid	Mid	Mid	Max	Mid	Max	Max	Max	Mid	Max
Min	Max	Max	Min	Mid	Max	Max	Min	Max	Max	Max	Max

$$RTT_{observed} = RTT_{observed} + 1 \quad (1)$$

$$\text{and if : } RTT > RTT_{observed} + d_{thresh} \quad (2)$$

$$\text{then : } RTT_{S\_late} = RTT_{S\_late} + 1 \quad (3)$$

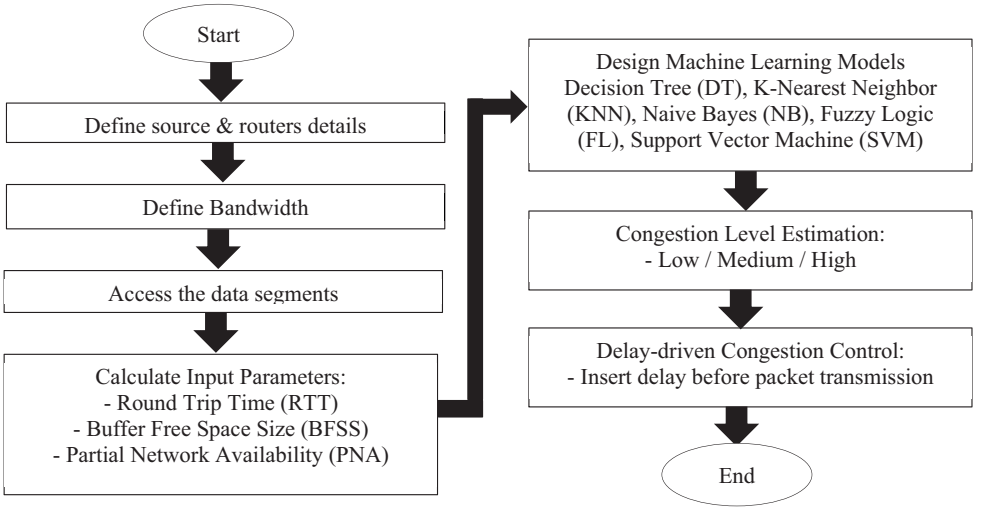

Figure 1. Flowchart of proposed research methodology

RTT_{min} *:* lowest limit of RTT. It represents the approximate value of propagation delay occurred in the network. $RTT_{observed}$: observed RTT. RTT_{late} is RTT times occurred as the exceeding to the RTT threshold. d_{thresh} represents delay threshold value. If K is the middle buffer size threshold, and the C represents the middle link bandwidth:

$$D_{thresh} = K / C \quad (4.1)$$

BFSS: Buffer Free Space Size, CL: Congestion level

Partial Network Availability (PNA): It represents the availability of the bandwidth of the network resources. It is used to indicate the spare capacity network. For a given link j, if C is data transmission capacity and cr is current traffic rate, the link spare capacity defined by, $sc_j = (C\text{-}cr)/C$. Hence, for 'w' number of links on the route from source to destination, PNA is defined as:

$$PNA = \min(sc_1, sc_2, ..., sc_w) \quad (4.2)$$

PNA is a feature that is easily computed and accessible network measurement applications.

Based on input parameters like Round-Trip Time (RTT), Buffer Free Space Size (BFSS), and Packet Number Acknowledgment (PNA), as shown in Table 2, the method described in this article suggests an intelligent decision-making mechanism that uses machine learning techniques to estimate the congestion level (CL). The decision rules state that when RTT is at its highest and both BFSS and PNA values are at their lowest, CL is categorized as high. This suggests that a high congestion situation is probably indicated by a large delay or a restricted buffer capacity. Under the case of high *CL* as per the equation (1), (2) and (3) additional wait time (delay) is inserted prior to send next packet. The inserted delay is incremented until the *CL* do not reaches medium ('Mid') level. According to the established equations (1), (2), and (3), the algorithm creates an artificial delay prior to the transmission of future packets when a significant level of congestion is detected. Up until the congestion level drops to a medium (Mid) condition, this delay is progressively increased. Four predictive models—Decision Tree (DT), K-Nearest Neighbor (KNN), Naive Bayes (NB), Fuzzy Logic (FL) and SVM framework—are developed and tested using MATLAB's machine learning toolbox in order to perform this adaptive decision process. These models are all regulated by the rule set displayed in Table 2.

3.1. Decision Tree (DT)

A DT consists of root node, internal decision nodes, branches, and terminal (leaf) nodes arranged in a hierarchical fashion. It is a supervised, non-parametric learning model that creates subsets, which ultimately lead to leaf nodes. These leaf nodes are responsible for final predictions. The learning process starts at the root, where data is gradually divided by assessing qualities at each internal node.

This design uses a clear and understandable logic flow to support methodical decision-making. Until the great majority of samples are properly identified, the model learning process partitions the data from top to bottom. Big trees lead to over fitting and increased data fragmentation. Pruning approaches are used to simplify the model by eliminating branches that have poor feature relevance in order to counteract this. Cross-validation is utilized to verify performance, and the CART (Classification and Regression Tree) method minimizes Gini impurity to identify the best splits. A value of 0 indicates a totally pure class distribution, and this measure represents the likelihood of misclassifying a randomly chosen case.

$$\text{Gini Impurity} = 1 - \sum_i (p_i)^2 \qquad (5)$$

The decision tree developed by using rule base of table 2 is shown in figure 2a. Here $\{x_1, x_2, x_3\}$ are input variable {*RTT, BFSS, PNA*} and the output is CL having three possible class label value as 1,2 or 3 i.e. LOW, MED or HIGH. The decision tree model developed in Statistics and Machine Learning Toolbox of MATLAB.

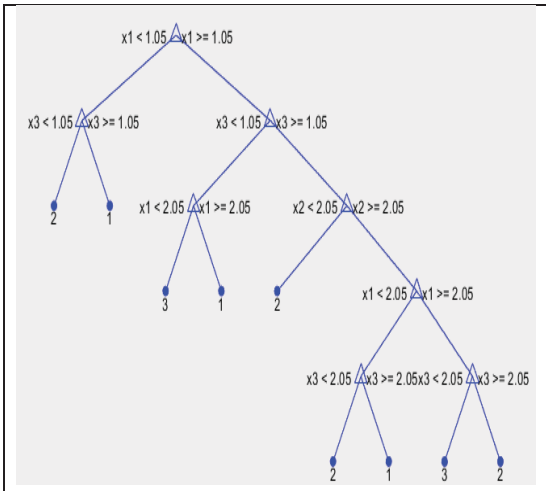

Figure 2a. Decision tree structure for DTBTCP

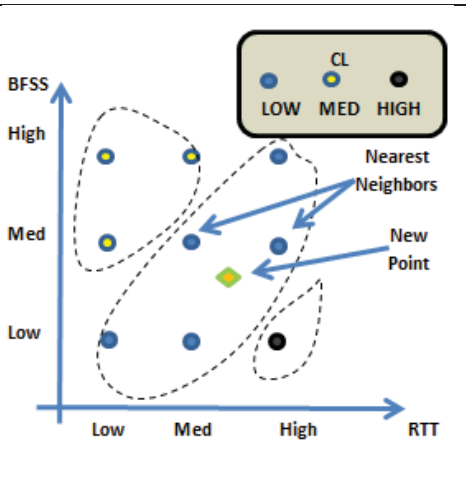

Figure 2b. k-NN structure for determining new data point congestion level

3.2. K-Nearest Neighbor (KNN)

A popular supervised learning method for categorizing new data according to how similar it is to existing cases is the algorithm. KNN is not aware about the distribution of the underlying data, it is a non-parametric approach. It works by storing the training dataset and making predictions at runtime. The K-NN method is used in this study to classify data, with an emphasis on deciding the congestion level (CL). The method is able to assign incoming queries to the most appropriate class based on their closeness to already labeled data. The CL is divided into three predetermined categories: low, medium, and high. For a new data point as the query with features input as x_1 (RTT), x_2 (BFSS) and x_3 (PNA), the K-NN decides the category to the query point belongs on the basis of its resemblance to training data records. The steps that are associated for learning of KNN decision model are given below (Srivastava and Banoudha, 2012):

1. Select the number of neighbors (K).
2. Calculation of the distance between K neighbors.
3. Choosing the K nearest neighbors.
4. Determination of the number of data points within a CL category among the K neighbors.
5. Putting the new data into category with the maximum number of neighbors.

3.3. Kernel Based Naive Bayes Classifier

A basic probabilistic model based on Bayes' theorem is called Naive Bayes (NB). It makes the assumption that characteristics are statistically independent in order to assess the likelihood of various classes. For example, the classifier may still predict output as "High" if BFSS is "Low" and RTT is "High". Even with small training data, Naive Bayes frequently produces trustworthy results. In order to improve non-parametric probability estimation, this paper employs a kernel-based version of

the NB classifier that incorporates kernel functions. In order to estimate the probability density functions, these kernel functions serve as weighting methods. It models underlying data distributions adaptively using non-parametric estimators that take into account the complete dataset. X as input feature set of RTT, BFSS and PNA as (x_i= x_1, $x_{2....}$, x_N) and C define set of CL level (class label) such that $c_j \epsilon$ {Low, Med, High}.

$$c = \max_{c_j \in C} P(c_j | x_1, x_2 \dots x_n) \quad (6.1)$$

$$P(x_1, x_2 \dots . x_n \mid c_j) = \prod_i P(x_i \mid c_j) \quad (6.2)$$

$$P(x_i \mid c_j) = \frac{1}{Nh} \sum_{v=1}^{N} K(x_i, x_{iii}) \quad (6.3)$$

$$K(a, b) = \text{kernel function} = \frac{1}{\sqrt{2\pi}} e^{\frac{(a-b)^2}{2h^2}} \quad (6.4)$$

$P(x_i|c_j)$ represents probability of CL equal to level x_i given that the input RTT, BFSS and PNA belongs to class label c_j estimated using training data [X | C]. Here Gaussian function kernel with zero mean and unity variance , N : number of the I/P data X belonging to class j which is equal c_j , x_{iii} is the feature value of the CL in the i^{st} position of the iii^{rd} input X = (x_{1i}, x_{2i} … x_{Ni}) in class j, and h defines width as a smoothing parameter. To accurately estimating the conditional probabilities, training dataset is used to optimize value of h.

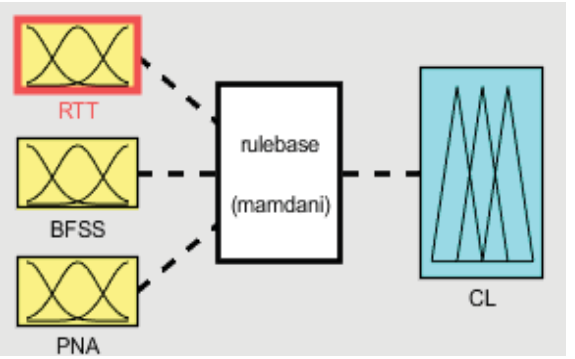

Figure 3. Fuzzy logic model for determination of congestion level

3.4 Fuzzy Logic

A useful machine learning approach, fuzzy logic (FL) is built on a rule-based framework that draws inspiration from human experience and reasoning. In this study, fuzzy algebra is used to build a FL system that estimates the degrees of network congestion. Figure 3 shows the input variables, RTT, BFSS, and PNA, indicated as X = {x1, x2, x3}. Gaussian membership functions are used to model the input and output variables. Every input value is allocated with varying degrees of confidence among the specified membership functions (Min, Mid, Max) in the fuzzy framework. A weighted sum of all relevant fuzzy rules outcome is calculated to arrive at the final output, which represents the most compromise-based or balanced answer (Sahay et al., 2012).

3.5 Support Vector Machine (SVM)

SVM is a supervised ML technique that finds the best hyperplane in a feature space to divide data points into various classes. Support vectors, or the margin between the closest data of opposing classes, are maximized using SVM. This approach works very well with high-dimensional datasets and maintains its dependability even when there are more features than samples. The concept of maximum margin classifiers is the conceptual foundation of SVM. SVM can efficiently convert non-linearly separable data into a higher-dimensional space with a linear boundary by utilizing kernel functions. SVM is still preferred in situations requiring interpretability, high speed processing, and reliable results with less training data, even if deep learning techniques have gained popularity recently. The rule base of table 2 given a training dataset D = $\{(x_i, y_i)\}_{i=1 \text{ to } n}$, where x_i : input vector (RTT,BFSS,PNA) & $y_i \in \{1,2,3\}$ as CL is class label, the SVM attempts to find weights 'w' and bias 'b' as vectors that define the decision boundary $f(x) = w^T x + b$. The optimization objective is to maximize the margin $2/\|w\|$ or equivalent to minimizing $\frac{1}{2}\|w\|^2$, subject to the constraint $y_i(w^T x_i + b) \geq 1$ for all training samples. To address the case where data is not linearly separable, the soft-margin SVM introduces slack variables $\xi_i \geq 0$, modifying the constraint to $y_i(w^T x_i + b) \geq 1 - \xi_i$, and incorporates a regularization parameter $C > 0$ to penalize misclassifications, leading to the objective function:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \qquad (7.1)$$

$$\text{subject to } y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0 \qquad (7.2)$$

For non-linear data distributions, SVM leverages the kernel to implicitly map the input data into a higher-dimensional feature space $\phi(x)$ without explicitly computing ϕ. Kernels 'K' that are commonly used are the linear kernel $K(x_i, x_j) = x_i^T x_j$, polynomial kernel $K(x_i, x_j) = (x_i^T x_j + c)^d$, and radial basis function (RBF) kernel $K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$, the γ is used to control the spread of the kernel.

4. RESULTS

The algorithm is implemented on MATLAB software using Instrument Control toolbox and Machine learning toolbox. The TCP protocol functions commands are used to simulate separate algorithms for server side and client side. Both the server and client models are run in parallel separately as two instances of MATLAB on same PC. The algorithm runs separately as execution of server and client side ports.

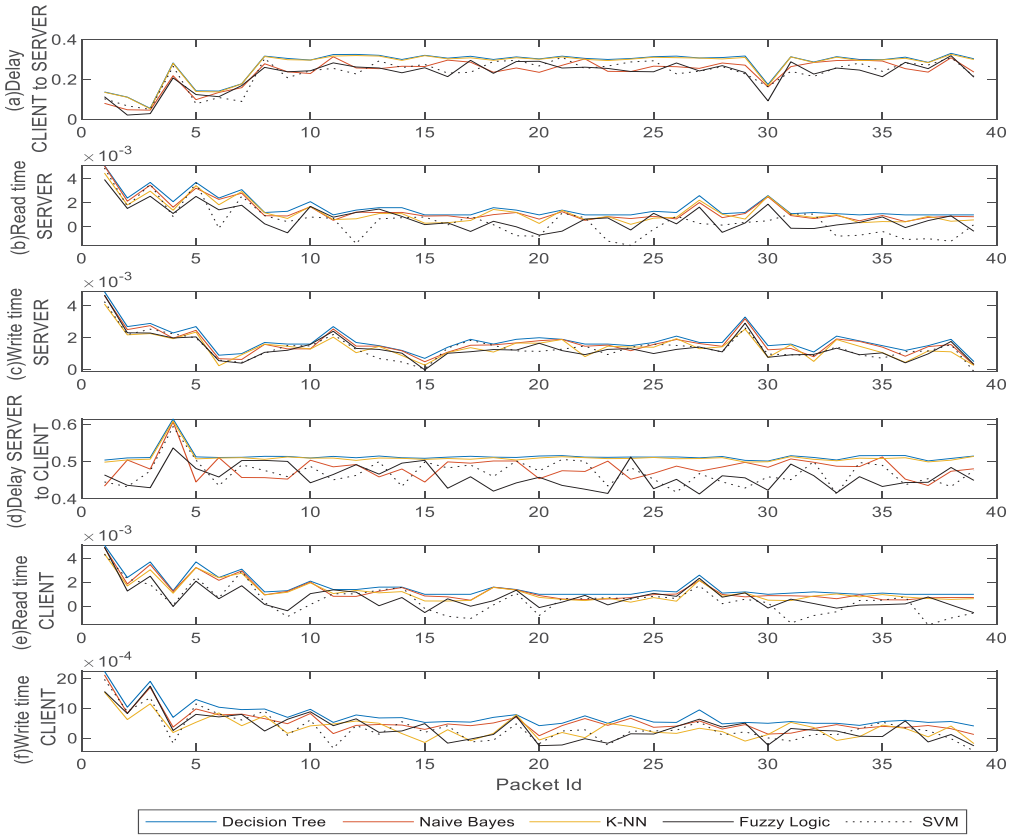

Figure 5 Plots showing performance as (a) Delay (b) Read time (d) delay server to client (e) read time client (f) write time client.

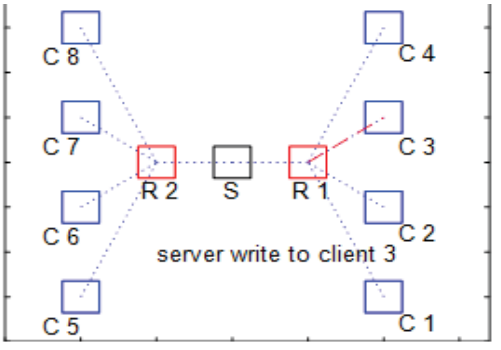

Figure 4. Data flow instant during writing data from server to client 3.

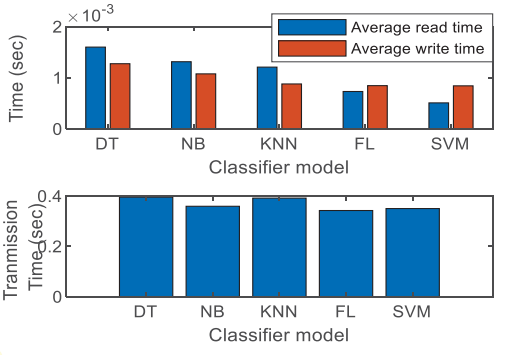

Figure 6. Average read /writes time (top) transmission time (bottom)

The topology of single dumbbell standard link (figure 4) is simulated by executing read write operation running the server/client roles of different clients, *n is* representing the number of clients (*Client* 1 to 4*)*

at left hand side and (*Client* 5 *to 8)* at right hand side) that are sending or receiving data via the server S by sharing the common single bottleneck link with buffer size of one BDP. Figure 4 showing an arbitrary screenshot of event during write operation indicating packet transmission from server to client 3. Similar type of read/write operation is executed randomly in between different clients through the router R1 and R2. The machine learning models are created from machine learning toolbox and TCP/IP interface under instrumentation toolbox of MATLAB software. The weather data of pressure value imported from csv file is randomly transmitted and received from the single server dumb bell shape link. Total 100 packets transmitted through the interface and the detection accuracy of congestion level of various ML models under multiple parameters setting of cross validation/ hold out validation are evaluated and shown (Table 3). The parameter considered during ML models learning are k-fold validation {2,3,4 and 5} folds and percentage of holdout validation (25%, 35%, 45% and 50%). The performance is determined in accuracy, observations per seconds (obs/sec), misclassification cost and time taken in training the ML model. Performance evaluated under different parameters based design of five ML models known as decision tree (DT), Naïve Bayes classifier (NB), K-nearest neighbor (KNN), Fuzzy logic (FL) and support vector machine (SVM). Performance in terms of accuracy is obtained higher under holdout validation compared to k-fold validation cases. SVM is showing maximum accuracy of 70.4% at 45% training data kept at hold out for validation. It is showing that ML models performance is not best at default parameters. All ML methods require parameter set tuning to attain best performance. Lowest misclassification cost is also exhibited by SVM at respective highest accuracy hold out value. Decision tree takes low training time but possess very low accuracy hence not suitable for this process. The SVM at highest accuracy also shows high value of observation per second and low time taken in training the model. Performance analysis for transmission phase of data is shown in figure 5 as the graphs demonstrating (a) Delay time (b) Read time (d) delay time server to client (e) read time at client side (f) write time at client side. The figure 6 shows the average read / write time (top) transmission time (bottom) during the transmission of data packets between different clients through the common server. SVM is capable giving least delay in all plots.

The congestion level represented as size of congestion window is observed with respect to time as shown in figure. It may be observed that the congestion window size in terms of packets is lying in between 900 to 1000 for proposed DTBTCP algorithm. This value is observed to be lowest on comparison to TCP, TCP FIT and Elastic TCP. Hence it shows that proposed updates in TCP using decision tree approach is helpful in reducing congestion compared to other state of art schemes [23, 24]. The trade-off observed is that DDCCP improves fairness by moderating the sending rate more conservative way, which may slightly limits the peak bandwidth efficiency under ideal conditions. At the same time, the moments when cache hits occur, bandwidth efficiency is becoming higher, but this temporarily reduce the fairness in between data flows with unequal cached-path performance. Thus, DDCCP manages a dynamic balance: fairness improves under congestion, while bandwidth efficiency improves under high cache availability—resulting in a controlled and predictable trade-off between the two.

Table 3. Results summary for applying different machine learning models

		k-fold	Accuracy	misclassification	obs/sec	train
Decision Tree	cross validation	2	33%	18	6200	0.839
		3	40.7%	16	4000	0.842
		4	33%	18	3400	0.857
		5	44%	15	2600	0.901
	hold out validation	25%	50%	3	3300	0.93
		35%	**55.6%**	4	3000	1.48
		45%	50%	6	3900	0.95
		50%	46%	7	2600	1.187
Kernel Naive Bayes	cross validation	2	44.4%	15	2300	0.966
		3	59%	11	1500	1.136
		4	59.3%	11	1400	1.141
		5	59%	11	890	1.157
	hold out validation	25%	66.6%	2	680	1.176
		35%	**66.7%**	3	2200	1.218
		45%	58%	5	1400	1.274
		50%	54%	6	3500	1.202
KNN	cross validation	2	63%	10	2100	2.256
		3	51.9%	16	1500	2.341
		4	59.3%	11	1500	1.378
		5	55.6%	12	1400	1.329
	hold out validation	25%	**66.7**%	2	2500	1.476
		35%	66.5%	3	2800	1.432
		45%	58.3%	5	4200	1.422
		50%	61.5%	5	5100	1.4112
Fuzzy Logic	cross validation	2	53%	10	1100	2.526
		3	61.9%	9	2500	1.491
		4	68.3%	2	5200	1.013
		5	51.6%	7	4400	1.025
	hold out validation	25%	56.7%	4	1500	1.261
		35%	46%	7	3800	1.132
		45%	**69.3**%	4	5100	1.312
		50%	51.5%	6	3200	1.041
SVM	cross validation	2	64.2%	43	36000	0.758
		3	50	60	6100	0.760
		4	51.7	58	6400	0.756
		5	54.2	55	5200	0.786
	hold out validation	25%	50	5	6400	0.885
		35%	50	2	4900	0.889
		45%	**70.4**	6	28000	0.821
		50%	50.0	3	11000	0.859

In an unreliable wireless environment, delay as the congestion indicator lead to misinterpretation of network conditions because variations in delay may also occur due to factors like channel fading, interference, mobility, and medium-access contention. This abrupt fluctuation in delay may cause the congestion control mechanism to incorrectly infer congestion and unnecessarily reduce its sending rate, leading to underutilization of available bandwidth. High variation in RTT reducing the reliability of delay thresholds such as RTT_{min} and d_{thresh}; these need frequent recalibration, making delay-based decisions less stable.

Novelty: This work introduces a delay-driven congestion control framework enhanced with machine learning models that intelligently estimate congestion levels using RTT, buffer free space, and partial network availability. Unlike prior methods, it enables per-path autonomous control, faster convergence, and efficient exploitation of cached data, ensuring optimal bandwidth utilization with

reduced packet loss.

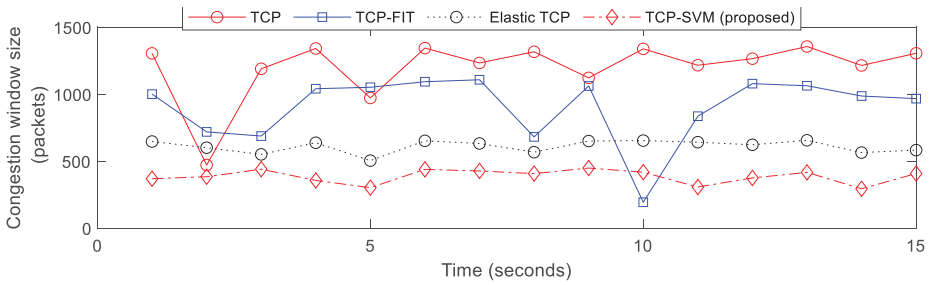

Figure 7. Comparison of congestion level with respect to time

5. CONCLUSION

A multi-source and multipath communication simulation system has been developed to tackle the complex issues of transport protocol design. This study introduces machine learning (ML) models-enhanced Network Utility Maximization (NUM) framework that facilitates delay-oriented congestion minimization techniques. The suggested ML-based protocol provides a more sophisticated option to traditional loss-based methods by functioning in a path-specific transport environment. This approach allows users to more accurately predict network congestion by depending on delay driven congestion control mechanism. Additionally, a thorough set of congestion estimation rules has been developed, which has several immediate advantages: it makes use of the high-bandwidth benefits of cached data, enables to respond with congestion more effectively, and guarantees quicker convergence in terms of fairness and bandwidth utilization. The MATLAB environment has been used to completely implement and evaluate the protocol through simulations. Building a real-time transmission model and expanding the system's functionality to accommodate more complex multipath communication scenarios will be the main goals of future improvements. The practical applicability of the DDCCP is its compatibility with dynamic, caching-enabled Content-Centric Networking environments, where traditional congestion control methods fail due to unstable paths, lack of endpoint visibility, and inconsistent cache behavior. This makes it feasible to deploy in overlay networks, wireless multi-hop environments, IoT domains, and any architecture where internal router metrics are inaccessible. This enhances performance in scenarios where paths exhibit different cache-hit ratios or sudden bottleneck shifts, such as in distributed content delivery or multi-access edge computing. DDCCP improves overall bandwidth utilization, reduces packet loss, and achieves faster fairness convergence, making it highly practical for real-time content retrieval and multimedia transmission within CCN architectures.

In terms of computational overhead, the machine-learning–assisted congestion estimator introduces only minimal processing cost and is suitable for real-time use. Although five ML models—Decision Tree, Naive Bayes, k-NN, Fuzzy Logic, and SVM—were evaluated, their training processes occur offline and thus do not burden the live network. Runtime inference is lightweight because the input parameters (RTT, BFSS, PNA) are simple scalar values, and congestion classification is executed

only when ACKs are received. Among all models, SVM demonstrated both the highest accuracy and the fastest inference speed, processing up to tens of thousands of observations per second, indicating that ML computation does not hinder DDCCP responsiveness. The experimental results further confirm that ML overhead does not negatively affect delay, read/write time, or packet transmission time; instead, the SVM-based implementation consistently achieved the lowest delay and fastest throughput across all test conditions.

6. REFERENCES

Zhang, L., Afanasyev, A., Burke, J., Jacobson, V., Claffy, K.C., Crowley, P., Wang, L., Zhang, B., 2014, Named data networking. ACM SIGCOMM Comput. Commun. Rev. **44**, 66–73.

Bobrov E., Panchenko S., Lu H., 2023, Fast Evaluation of the Quality of Wireless Systems using Machine Learning Algorithms. International Journal of Artificial Intelligence, **21,** 2, 39-60.

R. Mehta, 2021, Cross-layer Analysis of Prediction Accuracy in Ad-hoc networks based on Multivariate Logistic Regression. International Journal of Artificial Intelligence, **19,** 2, 1-19.

Wischik, D., Raiciu, C., Greenhalgh, A., Handley, M., 2011, Design, implementation and evaluation of congestion control for multipath TCP. Proc. 8th USENIX Conf. Networked Systems Design and Implementation, 99–112.

Zheng, L., Zhang, X., Zhang, S., Chen, X., 2021, Research on multi-path network in cloud computing based on SCTP. 8th IEEE Int. Conf. Cyber Security and Cloud Computing / 7th IEEE Int. Conf. Edge Computing and Scalable Cloud, 30–35.

Chen, Q., Xie, R., Yu, F.R., Liu, J., Huang, T., Liu, Y., 2016, Transport control strategies in named data networking: a survey. IEEE Commun. Surv. Tutorials **18**, 2052–2083.

Ye, Y., Lee, B., Flynn, R., Murray, N., Fang, G., Cao, J., Qiao, Y., 2018, PTP: path-specified transport protocol for concurrent multipath transmission in named data networks. Comput. Netw. **144**, 280–296.

Bai, H., Li, H., Que, J., Smahi, A., Zhang, M., Chong, P.H.J., Lu, P., 2024, QSCCP: a QoS-aware congestion control protocol for information-centric networking. IEEE Trans. Netw. Serv. Manag.

Khanal, D.P., 2024, Multi-commodity dynamic flow problems with intermediate storage and varying transit times. PhD Thesis, Tribhuvan University, Institute of Science and Technology, Kathmandu, Nepal.

Ould Khaoua, A.S., Boukra, A., Bey, F., 2024, On efficient interest forwarding in named data networks over IEEE 802.15.4: a comprehensive performance evaluation. Cluster Comput. **27**, 8065–8097.

Schneider, K., Yi, C., Zhang, B., Zhang, L., 2016, A practical congestion control scheme for named data networking. Proc. 3rd ACM Conf. Information-Centric Networking, 21–30.

Moiseenko, I., Oran, D., 2017, Path switching in content centric and named data networks. Proc. 4th ACM Conf. Information-Centric Networking, 66–76.

Gao, Q., Qiu, Q., Yin, Y., Zhang, G., 2024, A fair flow congestion control for content centric networking. 12th Int. Conf. Advanced Cloud and Big Data, 260–265.

Zhang, F., Zhang, Y., Reznik, A., Liu, H., Qian, C., Xu, C., 2014, A transport protocol for content-centric networking with explicit congestion control. Int. Conf. Comput. Commun. Netw., 1–8.

Mahdian, M., Arianfar, S., Gibson, J., Oran, D., 2016, MIRCC: multipath-aware ICN rate-based congestion control. Proc. 3rd ACM Conf. Information-Centric Networking, 1–10.

Bai, H., Li, H., Que, J., Zhang, M., Chong, P.H.J., 2022, DSCCP: a differentiated service-based congestion control protocol for information-centric networking. IEEE Wireless Commun. Netw. Conf., 1641–1646.

Rozhnova, N., Fdida, S., 2014, An extended hop-by-hop interest shaping mechanism for content-centric networking. IEEE Global Commun. Conf., 1–7.

Chen, J., Wang, Y., Huang, X., Xie, X., Zhang, H., Lu, X., 2022, ALBLP: adaptive load-balancing architecture based on link-state prediction in software-defined networking. Wireless Commun. Mobile Comput. 2022, 8354150.

Zhao, Y., Wang, C., Deng, H., 2025, Hop-by-hop multi-topology traffic engineering for inter-datacenter WANs. IEICE Trans. Commun.

Askar, N.A., Habbal, A., Alden, F.Z., Wei, X., Alaidaros, H., Guo, J., Yu, H., 2023, Forwarding strategies for named data networking based IoT: requirements, taxonomy, and open research challenges. IEEE Access **11**, 78363–78383.

Udugama, A., Zhang, X., Kuladinithi, K., Goerg, C., 2014, An on-demand multi-path interest forwarding strategy for content retrievals in CCN. IEEE Network Operations and Management Symp., 1–6.

Posch, D., Rainer, B., Hellwagner, H., 2016, SAF: stochastic adaptive forwarding in named data networking. IEEE/ACM Trans. Netw. **25**, 1089–1102.

Srivastava, A., Banoudha, A., 2014, Techniques of visualization of web navigation system. Int. J. Res. Dev. Appl. Sci. Eng. **6**.

Sahay, S., Banoudha, A., Sharma, R., 2012, On the use of ANFIS for ground water level forecasting in an alluvium area. Int. J. Res. Dev. Appl. Sci. Eng. **2**, 1.

He, B., Wang, J., Qi, Q., Sun, H., Liao, J., Lu, L., Han, Z., 2023, Learning-based real-time transmission control for multi-path TCP networks. IEEE Trans. Cogn. Commun. Netw. **9**, 1353–1369.

Boudi, A., Loudini, M., 2024, An advanced scheme for queue management in TCP/IP networks. arXiv:2402.04818.