*Journal of Communications, vol. 21, no. 3, 2026*

# Neural Adaptive Bitrate Streaming for Video Delivery over Named Data Networking: A Comparative Evaluation

Hasan Nur Arifin, Nana Rachmana Syambas, and Tutun Juhana*

School of Electrical Engineering and Informatics, Institut Teknologi Bandung, Bandung, Indonesia
Email: asanazam@gmail.com (H.N.A.); nana@stei.itb.ac.id (N.R.S.); tutun@stei.itb.ac.id (T.J.)
*Corresponding author

***Abstract*****—The rapid growth of online video traffic has intensified the need for Adaptive Bitrate (ABR) mechanisms capable of maintaining stable Quality of Experience (QoE) under diverse network conditions. While neural-based ABR approaches have shown promising results in IP-based streaming, their applicability to Named Data Networking (NDN) remains underexplored. This paper presents an experimental evaluation of a lightweight neural-based ABR scheme for NDN video streaming. Five representative ABR algorithms—Buffer-Based, Rate-Based, Simple, Hybrid, and Neural—are implemented using a customized Shaka Player integrated with the NDNts library. Experiments are conducted on an NDN-enabled testbed under low (2–3 Mbps), medium (5 Mbps), and high (10–15 Mbps) bandwidth conditions. Performance is evaluated using QoE-related metrics including startup latency, buffering behavior, dropped frames, and an overall QoE score. The results demonstrate that the proposed Neural ABR consistently achieves lower startup latency, reduced buffering, and higher QoE compared to conventional approaches across all bandwidth scenarios. These findings indicate that lightweight neural models can effectively enhance playback stability and QoE in NDN-based streaming systems, providing a practical bridge between IP-centric ABR designs and emerging content-centric networking architectures.**

***Keywords*****—Named Data Networking (NDN), Adaptive Bitrate (ABR), neural networks, video streaming, Quality of Experience (QoE)**

## I. INTRODUCTION

In the digital era, video streaming has become the dominant contributor to Internet traffic. Recent reports indicate that video services account for nearly 80% of global mobile data traffic, a proportion expected to grow further with the deployment of 5G and beyond [1]. This rapid increase places significant strain on conventional IP-based delivery mechanisms, where bandwidth variability, latency fluctuations, and congestion often lead to unstable Quality of Experience (QoE). As user expectations continue to rise, more adaptive and future-ready networking paradigms are required to sustain seamless video playback.

Named Data Networking (NDN) has emerged as a promising candidate for next-generation Internet architecture by shifting communication from a host-centric to a data-centric model [2]. By retrieving content based on names rather than host addresses, NDN inherently supports features such as in-network caching and flexible forwarding. Prior studies have shown that these properties can improve delivery efficiency and scalability for video streaming applications compared to traditional TCP/IP-based systems [3, 4]. QoE-aware mechanisms combined with caching strategies have further demonstrated potential in reducing rebuffering and bitrate instability in content-centric networking environments [5].

Despite these advantages, the performance of video streaming over NDN remains strongly influenced by the Adaptive Bitrate (ABR) strategy adopted at the client side. Conventional ABR approaches, including throughput-based, buffer-based, and hybrid algorithms, are widely used due to their simplicity but often struggle under dynamic network conditions [6, 7]. These limitations are particularly evident in NDN environments, where data retrieval and congestion dynamics differ from those of IP-based networks, potentially leading to suboptimal bitrate decisions and degraded QoE.

To overcome the limitations of heuristic ABR schemes, recent research has explored machine learning–based approaches for adaptive streaming. Neural network and reinforcement learning–based ABR methods have achieved notable performance gains in IP-based systems by capturing complex relationships between network states and bitrate adaptation decisions [8, 9]. In parallel, several NDN-specific rate adaptation mechanisms have been proposed to improve streaming efficiency under heterogeneous conditions [10, 11]. However, many existing solutions either rely on rule-based heuristics or are designed around IP-centric assumptions, limiting their practicality and generalizability in real-world NDN streaming deployments.

Motivated by this gap, this paper presents an experimental evaluation of a lightweight neural-based

---

Manuscript received October 9, 2025; revised December 17, 2025; accepted January 27, 2026; published May 13, 2026.

doi:10.12720/jcm.21.3.341-350

ABR scheme for NDN video streaming. Rather than introducing a complex or computationally intensive model, the proposed approach focuses on a simple feedforward neural network suitable for real-time deployment and compatible with existing NDN application frameworks. The main contributions of this work are summarized as follows:

1) Lightweight Neural ABR Design: A feedforward neural network is developed to perform adaptive bitrate selection based on empirically derived bandwidth–bitrate mappings.
2) Practical NDN Integration: The proposed ABR scheme is implemented in a customized Shaka Player integrated with the NDNts library, enabling adaptive video streaming over an NDN-enabled testbed.
3) Comparative Experimental Evaluation: Five ABR algorithms—Simple, Rate-Based, Buffer-Based, Hybrid, and Neural—are evaluated under low, medium, and high bandwidth conditions using standard QoE metrics.
4) QoE-Oriented Performance Analysis: Experimental results demonstrate that the Neural ABR improves playback stability and achieves higher overall QoE compared to baseline approaches, particularly under low and moderate bandwidth scenarios.

The remainder of this paper is organized as follows. Section II reviews related work on adaptive streaming and NDN-based multimedia delivery. Section III describes the system architecture, neural model, and QoE formulation. Section IV presents the experimental results and discussion, and Section V concludes the paper with directions for future research.

## II. RELATED WORKS

### A. *Adaptive Bitrate Streaming Techniques*

Adaptive Bitrate (ABR) streaming enables dynamic adjustment of video quality in response to time-varying network conditions. Video content is encoded into multiple bitrate representations and segmented into fixed-length chunks, allowing the client to select an appropriate bitrate during playback based on network conditions and buffer status to maintain stable Quality of Experience (QoE).

Client-side ABR remains the most widely adopted deployment model due to its simplicity and independence from network infrastructure. Conventional ABR strategies are commonly categorized into throughput-based, buffer-based, and hybrid approaches. Throughput-based methods estimate near-future bandwidth using recent download rates, while buffer-based strategies rely on buffer occupancy to guide bitrate selection [12]. Hybrid approaches combine both indicators to balance responsiveness and playback stability [13].

Although these rule-based techniques are lightweight and easy to deploy, prior studies have shown that they often perform poorly under highly dynamic network conditions, such as wireless and mobile environments. Inaccurate bandwidth estimation and aggressive bitrate switching can lead to bitrate oscillations, frequent rebuffering, and degraded QoE [14]. These limitations highlight the need for more adaptive and robust bitrate selection mechanisms capable of handling complex and fluctuating network dynamics.

### B. *Neural Approaches to ABR*

To overcome the limitations of heuristic-based ABR strategies, recent research has increasingly explored machine learning–based approaches, particularly those leveraging neural networks. By learning nonlinear relationships between network conditions, buffer states, and QoE outcomes, neural ABR models enable more adaptive and context-aware bitrate decisions.

A representative example is Pensieve, which employs Deep Reinforcement Learning (DRL) to directly optimize ABR policies with respect to QoE objectives. Subsequent studies, such as Tiyuntsong, extend this framework by incorporating self-play reinforcement learning mechanisms to improve robustness across diverse network scenarios [15]. Beyond DRL, alternative neural architectures—including Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) such as LSTM and GRU—have been proposed to capture spatial and temporal dependencies in adaptive streaming sessions [16, 17]. In mobile and energy-constrained settings, approaches such as LL-GABR further integrate power-awareness into learning-based ABR frameworks [18].

Despite their performance benefits, many neural-based ABR solutions introduce significant training and inference overhead. Deep reinforcement learning models typically require extensive offline training and complex state representations, which limit their scalability and practicality for real-time deployment. Consequently, lightweight neural approaches, such as Feedforward Neural Networks (FNNs), have gained attention due to their lower computational complexity and fast inference. However, existing evaluations of these lightweight models remain largely confined to IP-based streaming environments, leaving their applicability to content-centric architectures such as NDN insufficiently explored [19]. These observations motivate the need for a lightweight neural ABR solution that can be practically evaluated within an NDN-based streaming framework.

### C. *Neural ABR in NDN Environments*

While neural-based ABR techniques have demonstrated strong performance in IP-based streaming systems, their application to Named Data Networking (NDN) remains relatively limited. NDN introduces architectural characteristics that fundamentally differ from IP networks, including content-centric naming, in-network caching, and multipath data retrieval, which significantly alter data delivery behavior [20]. As a consequence, conventional throughput estimation becomes less reliable, since video segments may be retrieved from heterogeneous paths or intermediate caches rather than a single end host. In addition, NDN specific factors such as Pending Interest Table (PIT) dynamics, cache behavior, and Interest Data round-trip delay can influence playback performance but are rarely considered in IP-oriented ABR designs.

Early efforts, such as PCLive, demonstrated the feasibility of real-time video streaming over NDN using heuristic rate adaptation mechanisms [21]. Other studies have applied machine learning techniques to broader NDN problems, including routing optimization and congestion control [22]. However, research specifically targeting client-side neural ABR mechanisms for NDN-based video streaming remains scarce. Existing approaches largely rely on rule-based adaptation or focus on multicast-oriented delivery, leaving the practical evaluation of neural ABR strategies in NDN environments insufficiently explored.

Motivated by this gap, this work presents an experimental evaluation of a lightweight neural-based ABR scheme within an NDN video streaming framework. A feedforward neural network is integrated into a customized Shaka Player using the NDNts library, enabling real-time bitrate selection based on observed bandwidth and buffer conditions while maintaining low computational overhead. Unlike prior ML-based ABR solutions primarily designed for IP networks, this study provides a systematic comparison of neural ABR against conventional approaches in an NDN-enabled testbed, highlighting its potential to improve playback stability and overall QoE.

To clearly position this work within the existing literature, Table I summarizes the evolution of ABR approaches, ranging from conventional rule-based algorithms to ML-driven techniques in IP-based systems, and situates the proposed lightweight neural ABR within the context of NDN-based video streaming.

TABLE I. Comparative Overview of ABR Approaches

| Approach | Representative Methods | Strengths | Limitations | Deployment Context |
|---|---|---|---|---|
| Conventional ABR | Throughput-based [12], Buffer-based [12] Hybrid [13] | Simple, widely adopted, low overhead | Bitrate oscillation, poor estimation under variability, frequent rebuffering [14] | IP-based streaming |
| ML-based ABR (IP) | Pensieve (DRL) [8], Tiyuntsong (GAN-DRL) [15], LSTM/GRU-based [16, 17], LL-GABR [18] | Learns complex dynamics, improved QoE, reduced buffering | High training cost, heavy inference, limited scalability | IP-based adaptive streaming |
| Lightweight Neural ABR (IP) | Feedforward NN (FNN) [19] | Fast inference, low resource usage, suitable for mobile/edge devices | Limited temporal awareness, reduced robustness under rapid fluctuations | IP-based streaming |
| Neural ABR in NDN (This Work) | Proposed FNN-based ABR | Lightweight design, real-time bitrate selection, improved startup latency and buffering performance | Limited network scale; advanced NDN features not yet exploited | NDN-based video streaming |

## III. RESEARCH METHODS

### A. Research Design

This study adopts a quantitative experimental research design to evaluate the performance of five Adaptive Bitrate (ABR) algorithms: Buffer-Based, Rate-Based, Simple ABR, Hybrid, and the proposed Neural ABR within a Named Data Networking (NDN) based video streaming environment. The objective is to provide a controlled and fair comparison of ABR strategies under representative bandwidth conditions.

Experiments are conducted under three bandwidth profiles: low bandwidth (2–3 Mbps), moderate bandwidth (5 Mbps), and high bandwidth (10–15 Mbps). These ranges are commonly used as reference levels for Standard-Definition (SD), High-Definition (HD), and high-quality HD video streaming in both industry reports and prior adaptive streaming studies [23]. While the experimental setup does not capture all real-world impairments such as severe jitter or abrupt congestion events it reflects typical access-network conditions and enables consistent evaluation across heterogeneous bandwidth regimes.

User-perceived streaming performance is assessed using four Quality of Experience (QoE) related metrics: startup latency, buffering time, dropped frames, and effective bitrate. Startup latency measures initial playback responsiveness, buffering time captures playback interruptions, dropped frames indicate rendering stability, and effective bitrate reflects bandwidth utilization efficiency. Together, these metrics provide a comprehensive characterization of responsiveness, smoothness, stability, and quality. For comparative analysis, selected metrics are later integrated into a composite QoE formulation, as described in Section 3.4.

### B. System Topology and Architecture

The video streaming system is deployed over a Named Data Networking (NDN) environment and follows a client–server architecture, as illustrated in Fig. 1. A WebSocket channel is used for application-level coordination between the client and server, while the delivery of video content is performed exclusively through native NDN communication.

On the client side, a customized Shaka Player is employed for video playback. The default Adaptive Bitrate (ABR) Manager is extended to support multiple ABR algorithms, including Buffer-Based, Rate-Based, Hybrid, and the proposed Neural ABR. The player interfaces with the NDNts library, which provides the networking engine for managing Interest–Data exchanges and a fetching module for retrieving DASH media segments. This integration replaces conventional HTTP-based requests with name-based NDN communication while preserving compatibility with existing adaptive streaming formats.

On the server side, video content is published using the ndn6-file-server, which serves DASH media files from a designated directory under hierarchical NDN name prefixes (e.g., / NDN / testbed / video / belajar / playlist.mpd). The server responds to incoming Interest packets by returning the corresponding Data packets according to forwarding rules defined in the Forwarding Information Base (FIB).

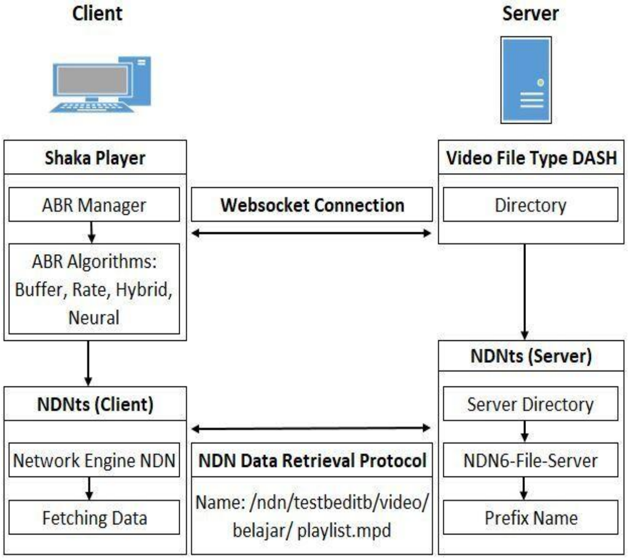

Fig. 1. System architecture of neural ABR with shaka player and nantes.

During playback, the client issues Interest packets for the manifest file and video segments. The requested Data packets are returned either directly from the server or from intermediate caches, when available, and are subsequently delivered to the Shaka Player for decoding and rendering. This architecture preserves the standard DASH workflow while enabling content-centric delivery over NDN, providing a practical and reproducible platform for evaluating adaptive bitrate algorithms in a next-generation networking environment.

### C. Player Data Flow

The internal workflow of the NDN-based adaptive streaming player is designed in a modular manner to enable flexible integration and fair comparison of multiple Adaptive Bitrate (ABR) algorithms. A customized Shaka Player coordinates the interactions among functional components responsible for manifest processing, bitrate adaptation, content retrieval, and video playback.

As illustrated in Fig. 2, the player workflow consists of four main modules: Manifest Parser, ABR Manager, Networking Engine, and Streaming Engine. The Manifest Parser processes the DASH manifest file (playlist.mpd) to extract essential metadata, including available bitrate representations and segment information. This metadata is forwarded to the ABR Manager to support bitrate selection and to the Networking Engine to initiate segment requests.

The ABR Manager performs adaptive bitrate selection and integrates four ABR strategies: Buffer-Based (BB), Rate-Based (RB), Hybrid, and the proposed Neural ABR allowing direct comparison under identical network conditions. Bitrate decisions are based on real-time observations of buffer occupancy and estimated throughput. While the BB, RB, and Hybrid approaches rely on predefined heuristic rules, the proposed Neural ABR employs a lightweight feedforward neural network to predict suitable bitrate levels with low computational overhead, making it suitable for real-time operation in an NDN-based streaming environment.

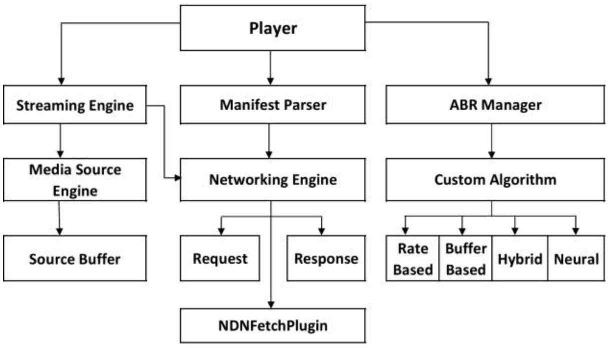

Fig. 2. Player data flow in the NDN-based streaming system.

Content retrieval is handled by the Networking Engine, which replaces conventional HTTP-based fetching with native NDN communication through a customized NDNFetchPlugin. Instead of issuing HTTP GET requests, the player requests video segments using NDN Interest packets and receives the corresponding content as Data packets via the request–response exchange. This design ensures that segment delivery follows a content-centric communication model consistent with NDN principles.

The Streaming Engine manages video playback by interfacing with the Media Source Engine (MSE) and maintaining the source buffer. Retrieved segments are decoded and appended to the buffer, which continuously feeds the playback pipeline. Effective buffer management helps reduce startup delay and mitigate playback stalls under varying bandwidth conditions.

Overall, the workflow follows a sequential and modular process: the Manifest Parser extracts segment information, the ABR Manager selects the target bitrate, the Networking Engine retrieves content through Interest–Data exchanges, and the Streaming Engine ensures continuous playback. This design preserves compatibility with standard DASH workflows while providing a practical and reproducible platform for evaluating neural-based adaptive bitrate streaming in Named Data Networking environments.

### D. Description of ABR Algorithm

This study evaluates five Adaptive Bitrate (ABR) algorithms implemented in a customized Shaka Player within an NDN-based streaming environment. Four algorithms—Buffer-Based, Rate-Based, Simple ABR, and Hybrid—serve as baseline heuristic approaches, while the Neural ABR represents the proposed learning-based method. All baseline algorithms are implemented using standard configurations without additional tuning to ensure a fair and reproducible comparison.

#### 1) Buffer-based ABR

The Buffer-Based ABR algorithm selects the video bitrate based solely on the client's buffer occupancy. Let B denote the instantaneous buffer level in seconds. The buffer state is linearly normalized into a discrete scale to guide bitrate selection).

$$L = \min(\max(B \times 10, 1), 10) \quad (1)$$

where $L$ represents the normalized buffer level. Based on this value, the buffer state is categorized into three regions: low (1–3), medium (4–7), and high (8–10). Lower buffer levels trigger conservative bitrate selection to prevent playback stalls, while higher levels allow bit rate increases to improve visual quality. This approach prioritizes playback stability but does not explicitly account for bandwidth variability.

#### 2) *Rate-base ABR*

The Rate-Based ABR algorithm adapts the bitrate according to the estimated network throughput. Throughput is computed from the most recently downloaded video segment as:

$$T = N/\Delta t \quad (2)$$

where N denotes the segment size and $\Delta t$ is the segment download time. The estimated throughput is then used to select the highest available bitrate that does not exceed the measured network capacity. While this method can react quickly to bandwidth changes, it is sensitive to short-term fluctuations and estimation errors.

#### 3) *Simple ABR (Shaka Default)*

The Simple ABR algorithm is the default strategy provided by the Shaka Player. It follows a conservative throughput-based policy by relying on long-term bandwidth estimation and safety margins when selecting the bitrate. This design prioritizes playback stability and minimizes quality oscillations, at the cost of slower responsiveness to rapid network variations.

#### 4) *Hybrid ABR*

The Hybrid ABR algorithm combines buffer-based decision logic with both short-term and long-term throughput estimation [24]. By jointly considering buffer occupancy and bandwidth trends, it aims to balance responsiveness and stability, reducing bitrate oscillations while maintaining efficient bandwidth utilization. This hybrid strategy represents a compromise between purely buffer-based and throughput-based approaches.

Overall, this baseline algorithms represent widely adopted heuristic ABR strategies in adaptive streaming systems. Their inclusion provides a standardized reference framework, ensuring that performance differences observed for the proposed Neural ABR can be attributed to learning-based adaptation rather than heuristic optimization or parameter tuning.

### E. *Neural ABR (Proposed Method)*

The proposed Neural ABR employs a lightweight Feedforward Neural Network (FNN) to perform adaptive bitrate selection based on real-time client-side observations. Unlike heuristic ABR schemes, the neural model learns nonlinear relationships between observed network conditions and bitrate decisions, enabling more stable adaptation in an NDN-based streaming environment while maintaining low computational complexity.

#### 1) *Decision flow and model operation*

Fig. 3 illustrates the decision flow of the proposed Neural ABR model. The architecture operates as follows.

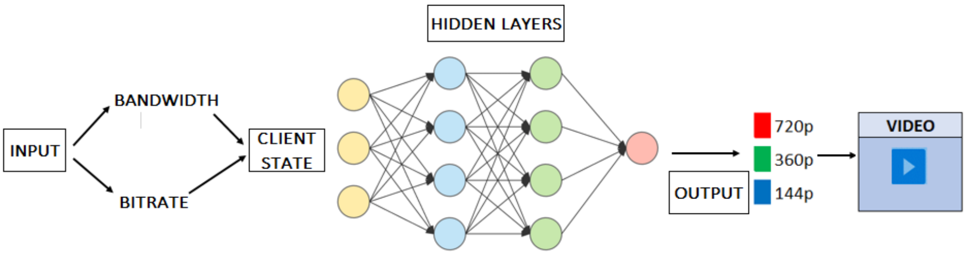

Fig. 3. Proposed neural ABR architecture.

First, the input layer receives two client-side features: the measured available bandwidth (Mbps) and the current playback bitrate state. These features capture the instantaneous network condition and the ongoing quality level.

Second, the inputs are combined into a feature vector and processed through one or more fully connected hidden layers with Rectified Linear Unit (ReLU) activation functions. This structure allows the model to capture nonlinear mappings between network dynamics and suitable bitrate selections without introducing excessive inference overhead.

Third, the output layer applies a softmax activation function to generate a probability distribution over a set of discrete bitrate classes (144, 360, 480, and 720). The bitrate corresponding to the highest probability is selected and applied for subsequent playback.

By formulating bitrate adaptation as a multi-class classification problem, the model enables fast inference and deterministic decision-making, making it suitable for real-time deployment in NDN environments where content retrieval delays may exhibit non-deterministic behavior.

#### 2) *Model training and dataset*

The neural model is trained using a dataset of approximately 12,000 state–action samples collected from multiple video streaming sessions conducted under controlled network conditions. Bandwidth profiles are varied to emulate low, moderate, and high capacity scenarios, ensuring coverage of representative streaming environments [25].

During each streaming session, samples are recorded at fixed decision intervals corresponding to bitrate selection events. Each sample captures a snapshot of the client-side streaming state and the associated bitrate decision. Although the number of streaming sessions is limited, time-step–level logging yields a sufficiently large dataset for supervised learning.

Each training sample consists of two input features: (i) the measured client-side bandwidth (Mbps) and (ii) the current playback bitrate state. The output is modeled as a multi-class label representing the selected bitrate–resolution level among four discrete classes (144, 360, 480, and 720).

Table II presents a representative subset of the training samples used to construct the supervised learning dataset for the Neural ABR model. Each sample maps observed client-side bandwidth conditions to a corresponding bitrate–resolution decision.

TABLE II. REPRESENTATIVE TRAINING SAMPLES FOR NEURAL ABR

| **Bandwidth (Mbps)** | **Target Bitrate (Mbps)** | **Resolution** |
|---|---|---|
| 1.1 | 0.5 | 144 |
| 1.5 | 0.5 | 144 |
| 2.0 | 1.5 | 360 |
| 2.5 | 1.5 | 360 |
| 2.9 | 1.5 | 360 |
| 3.5 | 2.5 | 480 |
| 3.9 | 2.5 | 480 |
| 4.0 | 3.5 | 720 |
| 4.5 | 3.5 | 720 |
| 5.0 | 3.5 | 720 |

The dataset is randomly divided into 80% training and 20% validation subsets. The Neural ABR model is trained as a multi-class classification problem using the following configuration:

1) Architecture: One input layer (two features), two fully connected hidden layers (16 and 32 neurons with ReLU activation), and one softmax output layer with four classes.
2) Loss Function: Categorical cross-entropy.
3) Optimizer: Adam with a learning rate of 0.001.
4) Batch Size: 32.
5) Training Epochs: Up to 50 epochs with early stopping based on validation loss.
6) Implementation Framework: TensorFlow/Keras.

This configuration is intentionally selected to balance learning capacity and computational efficiency. Given the limited input dimensionality and discrete decision space, the lightweight architecture enables fast inference and stable performance without requiring large-scale training data or complex model structures.

### 3) *Training results*

During training, the Neural ABR model achieves an average classification accuracy of 92.3% on the validation set, indicating effective learning of the mapping between observed bandwidth conditions, playback states, and bitrate classes. It is important to note that this accuracy reflects performance on the bitrate classification task rather than direct optimization of Quality of Experience (QoE).

The validation loss stabilizes after approximately 30 epochs, suggesting proper convergence of the training process. The application of early stopping prevents overfitting, as no significant divergence between training and validation loss is observed in later epochs.

Analysis of the confusion matrix shows that most misclassifications occur between adjacent bitrate classes, particularly between 360p and 480p. This behavior is expected due to overlapping bandwidth ranges and closely spaced bitrate thresholds. Importantly, the model avoids extreme misclassifications, such as selecting high-bitrate representations under constrained bandwidth or low-bitrate representations under favorable conditions. This property is critical for adaptive streaming systems, as extreme mispredictions are more likely to cause rebuffering or inefficient bandwidth utilization.

Overall, the training results demonstrate that the proposed neural model provides stable and consistent bitrate predictions under previously unseen bandwidth variations within the experimental setup. The combination of high classification accuracy, smooth convergence behavior, and controlled misclassification patterns supports the suitability of the proposed Neural ABR for real-time deployment in NDN-based adaptive video streaming environments.

### 4) *Quality of Experience (QoE) Evaluation*

To evaluate and compare the performance of different Adaptive Bitrate (ABR) algorithms, this study considers several Quality of Experience (QoE)–related metrics, including startup latency, buffering behavior, stream bandwidth, and frame quality. Among these metrics, startup latency, rebuffering behavior, and average bitrate are directly incorporated into the QoE formulation, while frame quality is reported as a complementary indicator of playback stability.

Startup latency represents the elapsed time between the playback request and the start of video rendering. Buffering behavior is characterized by the total accumulated stalling duration during playback, which has been widely recognized as one of the most disruptive factors affecting user experience. Stream bandwidth denotes the average effective bitrate achieved during playback and is positively correlated with delivered video quality. Frame quality is defined as the ratio of successfully decoded frames to the total number of frames and is used solely for stability analysis rather than as a direct QoE component.

Following prior adaptive streaming studies, this work adopts a linear QoE model that integrates the three dominant factors influencing user-perceived streaming quality: average bitrate, rebuffering behavior, and startup latency. The QoE score is computed as:

$$QoE = \alpha.\ Bitrate - \beta.\ RebufferingRatio - \gamma.\ Latency \quad (3)$$

where Startup latency is normalized to seconds before being used in the QoE computation. *Bitrate* denotes the average effective streaming bitrate (Mbps), *RebufferingRatio* is defined as the ratio between total buffering duration and total playback time, and *Latency* corresponds to the startup delay (seconds). The weighting coefficients α\alpha, β\beta, and γ\gamma reflect the relative importance of each QoE component. The rebuffering ratio is calculated as:

$$Rebuffering\ Ratio = \frac{Total\ Buffering\ TIme}{Total\ Pleayback\ Time} \quad (4)$$

In accordance with prior QoE modeling studies, buffering events are penalized more heavily due to their strong negative impact on perceived viewing quality. Accordingly, the weighting parameters are set to $\alpha = 1$, $\beta = 5$, and $\gamma = 1$. This configuration emphasizes playback continuity while maintaining balanced consideration of video quality and responsiveness.

It is important to note that the proposed QoE model is employed as a relative comparison metric rather than an absolute perceptual score. Consequently, negative QoE values may occur and are considered acceptable. The primary objective of this formulation is to enable consistent and transparent comparison of ABR algorithms under identical experimental conditions, rather than to approximate subjective Mean Opinion Score (MOS) values.

Table III shows that startup latency, buffering time, and stream bandwidth are directly incorporated into the QoE formulation, while dropped frames are reported as a complementary playback stability metric. Their impact is implicitly reflected through rebuffering behavior and is therefore not explicitly included in the QoE model to avoid double penalization.

TABLE III. QOE METRICS AND THEIR ROLES IN THE EVALUATION

| QoE Parameter | Definition | Role in Analysis | Included in QoE |
|---|---|---|---|
| Startup Latency | Time from playback request until video starts (ms) | Startup responsiveness indicator | Yes |
| Buffering Time | Total accumulated stalling during playback (s) | Used to compute rebuffering ratio | Yes |
| Stream Bandwidth | Average effective streaming bitrate during playback (Mbps) | Positive utility term for delivered video quality | Yes |
| Dropped Frames | Number of frames not successfully decoded | Complementary playback stability indicator | No |

## IV. RESULTS

### A. Experimental Setup and Conditions

This study evaluates five Adaptive Bitrate (ABR) algorithms—Buffer-Based, Rate-Based, Simple ABR, Hybrid, and the proposed Neural ABR—within a controlled Named Data Networking (NDN) video streaming testbed. All schemes are implemented using a customized Shaka Player integrated with the NDNts library, enabling native Interest/Data communication while preserving standard DASH-based adaptive streaming workflows.

To support reproducibility, the system is realized using a web-based architecture built on Next.js and Shaka Player. Quality of Experience (QoE) metrics, including startup latency, buffering time, dropped frames, and effective bitrate, are collected via the Shaka Player statistics API and processed into state representations for a lightweight machine learning model implemented in JavaScript (e.g., TensorFlow.js).

NDN validation is conducted in a single-node environment running Ubuntu Server 22.04/24.04 with lightweight hardware (1 CPU core, 2 GB RAM). The node operates NDN services through the NDNts library and delivers a single MPEG-DASH encoded MP4 video used consistently across all experiments, ensuring fair and repeatable evaluation without requiring complex multi-node NDN topologies.

#### 1) Network conditions

To Experiments are conducted under three representative bandwidth profiles to reflect practical video streaming environments:

- Low bandwidth: 2–3 Mbps
- Moderate bandwidth: 5 Mbps
- High bandwidth: 10–15 Mbps

These bandwidth ranges correspond to commonly reported thresholds for SD, HD, and high-quality video streaming in practical deployments and are consistent with prior adaptive streaming studies and industry recommendations. Evaluating multiple bandwidth regimes enables a comprehensive assessment of ABR adaptability, stability, and robustness under both constrained and favorable network conditions. The network dynamics are emulated using the Mahimahi network emulator to ensure controlled and repeatable bandwidth profiles [26].

#### 2) Measured performance metrics

Performance evaluation focuses on objective, client-side metrics that directly reflect user-perceived streaming quality. For each playback session, the following metrics are collected:

1) Startup Latency (ms): The elapsed time between the playback request and the start of video rendering.
2) Buffering Time (s): The total accumulated duration of playback interruptions caused by buffer underruns.
3) Dropped Frames (count): The number of frames not successfully decoded during playback, indicating playback stability.
4) Average Effective Bitrate (Mbps): The mean bitrate achieved during playback, reflecting delivered video quality and bandwidth utilization efficiency.

All metrics are collected at the client side using Shaka Player's built-in statistics and logging mechanisms. Each experiment is repeated across multiple playback sessions, and the reported results represent averaged values to ensure consistency, reliability, and reproducibility.

### *B. Quantitative Results*

Table IV summarizes the measured performance of all evaluated Adaptive Bitrate (ABR) algorithms under low (2–3 Mbps), moderate (5 Mbps), and high (10–15 Mbps) bandwidth conditions. The reported metrics—startup latency, buffering time, and dropped frames—capture key aspects of playback responsiveness and stability.

TABLE IV. PERFORMANCE OF ABR ALGORITHMS UNDER DIFFERENT BANDWIDTH CONDITIONS

| **Algorithm** | **BW (Mbps)** | **Latency (ms)** | **Dropped Frames** | **Buffering Time (s)** |
|---|---|---|---|---|
| Buffer-Based | 2–3 | 136 | 21 | 3.11 |
|  | 5 | 132 | 0.9 | 0 |
|  | 10–15 | 138 | 1.4 | 0.3 |
| Rate-Based | 2–3 | 166 | 2.0 | 5.81 |
|  | 5 | 151 | 2.6 | 0 |
|  | 10–15 | 144 | 1.1 | 0.2 |
| Simple ABR | 2–3 | 161 | 0 | 6.63 |
|  | 5 | 128 | 0.8 | 0 |
|  | 10–15 | 134 | 0.9 | 0.4 |
| Hybrid ABR | 2–3 | 64 | 0 | 5.81 |
|  | 5 | 115 | 0 | 0 |
|  | 10–15 | 121 | 0.3 | 0.2 |
| Neural ABR | 2–3 | 24 | 0 | 0 |
|  | 5 | 107 | 0 | 0 |
|  | 10–15 | 113 | 0.2 | 0.1 |

### *C. Quantitative Analysis*

Under low bandwidth conditions (2–3 Mbps), the proposed Neural ABR demonstrates a clear performance advantage over all baseline algorithms. It achieves the lowest startup latency (24 ms) and completely eliminates buffering and frame drops, indicating robust adaptation under constrained network capacity. In contrast, Buffer-Based and Simple ABR suffer from substantial playback interruptions, with buffering durations exceeding 3 s and 6 s, respectively, reflecting limited responsiveness to rapid bandwidth fluctuations.

At moderate bandwidth (5 Mbps), buffering events are largely eliminated across most algorithms. However, Rate-Based ABR still exhibits noticeable frame drops, suggesting sensitivity to short-term throughput estimation errors. In comparison, Neural ABR maintains stable playback with zero buffering and zero dropped frames, while also preserving low startup latency.

Under high bandwidth conditions (10–15 Mbps), all ABR schemes benefit from increased network capacity. Nevertheless, Neural ABR continues to provide the most stable overall performance, combining consistently low startup latency, negligible buffering (0.1 s), and minimal frame loss. These results indicate that the proposed neural approach adapts effectively across a wide range of bandwidth regimes without introducing instability under favorable network conditions.

### *D. QoE Evaluation Using Linear Model*

To objectively assess user-perceived streaming quality, a linear Quality of Experience (QoE) model is employed, as defined in Section III-D. The formulation integrates three dominant factors influencing QoE [27]: Average bitrate, rebuffering behavior, and startup latency. and is expressed as: $\text{QoE}= \alpha.\ \text{Bitrate} - \beta\cdot \text{RebufferingRatio} - \gamma\cdot\text{Latency}$ where the weighting coefficients are set to $\alpha = 1$, $\beta = 5$, and $\gamma = 1$, emphasizing playback continuity in accordance with prior adaptive streaming studies (see Table V).

TABLE V. QOE SCORE COMPARISON OF ABR ALGORITHMS

| **Algorithm** | **Average QoE Score** |
|---|---|
| Buffer-Based | 9.72 |
| Rate-Based | 9.85 |
| Simple ABR | 9.60 |
| Hybrid ABR | 10.90 |
| Neural ABR | 11.88 |

#### *1) QoE analysis*

Neural ABR achieves the highest overall QoE score, primarily due to its ability to minimize startup latency and rebuffering while maintaining competitive bitrate levels. Although Hybrid ABR performs reasonably well at higher bandwidths, its buffering penalties under constrained conditions reduce its overall QoE. In contrast, conventional heuristic-based algorithms are consistently penalized by buffering and latency, highlighting the advantage of learning-based adaptation in dynamic NDN environments.

#### *2) Comparative analysis*

The experimental results reveal clear trade-offs among the evaluated ABR strategies. Buffer-Based ABR is highly sensitive to buffer depletion, resulting in significant frame loss under low bandwidth. Rate-Based ABR reacts quickly to bandwidth variations but suffers from instability due to inaccurate short-term throughput estimation. Simple ABR favors stability but incurs higher latency and buffering overhead. Hybrid ABR improves this balance but remains vulnerable under constrained bandwidth conditions.

Overall, Neural ABR consistently delivers the most robust performance across all bandwidth regimes, achieving low latency, near-zero buffering, and stable frame delivery. These findings confirm the effectiveness of lightweight neural adaptation for NDN-based video streaming under dynamic network conditions.

### *E. Visual Insights*

Figs. 4–6 provide visual comparisons of the key QoE-related metrics across different bandwidth conditions. These figures complement the quantitative results by highlighting performance trends and relative differences among ABR algorithms.

Fig. 4 shows that under low-bandwidth conditions (2–3 Mbps), the proposed Neural ABR achieves the lowest startup latency (24 ms), significantly outperforming heuristic-based approaches. Simple ABR exhibits the highest startup delay, indicating slower playback initialization under constrained bandwidth. At moderate bandwidth (5 Mbps), Neural ABR maintains superior responsiveness, while other algorithms experience higher delays. Under high bandwidth (10–15 Mbps), latency differences among algorithms become less pronounced; nevertheless, Neural ABR consistently remains among the lowest, demonstrating stable responsiveness across bandwidth regimes.

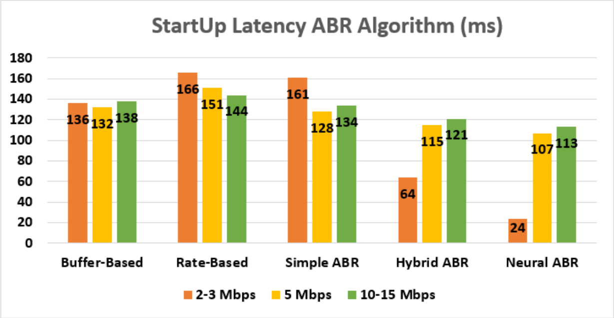

Fig. 4. Startup latency under different ABR algorithms.

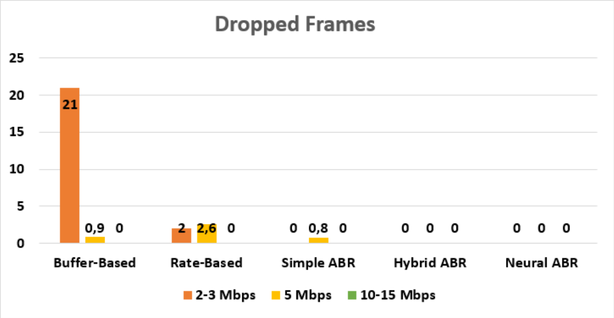

Fig. 5. Dropped frames under different ABR algorithms.

Fig. 5 illustrates playback stability in terms of dropped frames. At low bandwidth, Buffer-Based ABR suffers from the highest frame loss, whereas Hybrid and Neural ABR experience no dropped frames. At 5 Mbps, Rate-Based ABR shows noticeable frame drops, while the remaining algorithms exhibit negligible losses. These results indicate that learning-based and hybrid strategies provide improved rendering stability under bandwidth fluctuations.

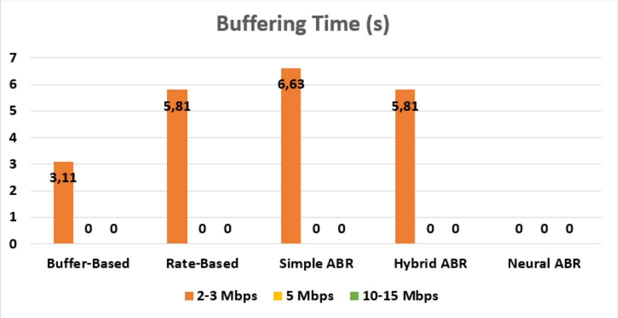

Fig. 6. Buffering time under different ABR algorithms.

Fig. 6 compares the buffering behavior of the evaluated ABR algorithms. Under low bandwidth, Simple ABR incurs the longest buffering duration, followed by Rate-Based and Hybrid ABR, while Buffer-Based ABR shows moderate buffering. In contrast, Neural ABR completely eliminates buffering. At moderate and high bandwidth levels, buffering is negligible for all algorithms, indicating that performance differences are most pronounced under constrained network conditions.

## V. CONCLUSION

This paper presents a comparative evaluation of five Adaptive Bitrate (ABR) algorithms: Buffer-Based, Rate-Based, Simple ABR, Hybrid, and the proposed Neural ABR within a Named Data Networking (NDN) video streaming framework. Experiments were conducted using a customized Shaka Player integrated with the NDNts library under low (2–3 Mbps), moderate (5 Mbps), and high (10–15 Mbps) bandwidth conditions.

The results demonstrate that the proposed Neural ABR, based on a lightweight Feedforward Neural Network (FNN), consistently outperforms baseline approaches across key Quality of Experience (QoE) metrics. It achieves the lowest startup latency, eliminates buffering and frame drops under low and moderate bandwidth conditions, and maintains only negligible buffering and frame loss at high bandwidth. As a result, Neural ABR attains the highest average QoE score among all evaluated algorithms. Compared with rule-based and hybrid schemes, the neural approach adapts more effectively to bandwidth variations while preserving low computational complexity.

Overall, these findings validate Neural ABR as an effective and scalable solution for adaptive video streaming in Named Data Networking (NDN) environments. The proposed approach is compatible with NDN's content-centric design and achieves stable performance across diverse bandwidth conditions without relying on computationally intensive reinforcement learning techniques.

Future work will extend the evaluation to multi-client scenarios and further exploit NDN-specific features such as multipath forwarding and in-network caching. In addition, the framework will be evaluated using more diverse network traces, larger-scale topologies, and comparisons with reinforcement learning–based ABR schemes under highly dynamic network conditions.

## CONFLICT OF INTEREST

The authors declare no conflict of interest.

## AUTHOR CONTRIBUTIONS

Hasan Nur Arifin primarily contributed to the conceptual design, analysis, software implementation, validation, and investigation of this study; Nana Rachmana Syambas and Tutun Juhana provided critical review of the content and were responsible for the overall oversight and management of the research presented in this paper; all authors had approved the final version.

## ACKNOWLEDGMENT

The author would like to thank the NDN Research Team from the ITB Telematics Lab for providing initial ideas and suggestions for the paper's initial writing structure.

## REFERENCES

[1] Cisco Systems, "Cisco Annual Internet Report (2018–2023)," White Paper, Cisco, 2022.

[2] A. Khelifi, J. Wang, M. A. K. Khan, and M. Atiquzzaman, "Information-centric networking and machine learning: A survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1676–1721, 2020.

[3] C. Ghasemi, H. Yousefi, and B. Zhang, "Internet-scale video streaming over Named Data Networking," *IEEE Network*, vol. 35, no. 5, pp. 174–180, 2021.

[4] X. Sun, J. Jiang, and X. Wang, "A stochastic optimization approach for dynamic adaptive streaming over Named Data Networking," *IEEE Transactions on Multimedia*, vol. 24, pp. 2575–2587, 2022.

[5] Y. Hayamizu, K. Goto, M. Bandai, and M. Yamamoto, "QoE-aware bitrate selection in cooperation with in-network caching for information-centric networking," *IEEE Access*, vol. 9, pp. 165059–165071, 2021.

[6] W. Li, M. Fayed, S. M. A. Oteafy, and H. S. Hassanein, "Quality of experience in ICN: Keep your low bitrate close and high-bitrate closer," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 557–571, Apr. 2021.

[7] D. Liu, Y. Wang, J. Liu, and H. Zhang, "Reinforcement learning-based dynamic adaptive video streaming for multi-client over Named Data Networking," in *Proc. IEEE International Conference on Communications (ICC)*, Seoul, South Korea, 2022, pp. 1–6.

[8] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with Pensieve," in *Proc. ACM SIGCOMM*, Los Angeles, CA, USA, 2017, pp. 197–210.

[9] A. Y. Li, J. Fang, J. Zhang, and X. Wang, "Improving ABR performance for short video streaming using multi-agent reinforcement learning with expert guidance," arXiv:2304.04637, 2023.

[10] F. Wu, W. Yang, P. Yang, J. Ren, Y. Zhang, F. Lyu, and X. Shen, "NDN-MMRA: Multi-stage multicast rate adaptation in Named Data Networking WLAN," *IEEE Transactions on Multimedia*, vol. 23, pp. 1–15, 2021.

[11] J. Shi, "NDNts: Named Data Networking libraries for the modern web," 2024.

[12] S. Akhshabi, L. Anantakrishnan, A. C. Begen, and C. Dovrolis, "An experimental evaluation of rate-adaptive video players over HTTP," *Signal Processing: Image Communication*, vol. 27, no. 4, pp. 271–287, 2012.

[13] A. A. Sani, P. Shetty, A. Razaghpanah, and P. Gill, "Characterizing and modeling video streaming performance: A measurement study of Hulu and Netflix," in *Proc. Passive and Active Measurement Conference (PAM)*, Berlin, Germany, 2017, pp. 1–12.

[14] L. De Cicco, S. Mascolo, and V. Palmisano, "Feedback control for adaptive live video streaming," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 2, pp. 40–47, 2011.

[15] Y. Mao, Y. Yu, and X. Lin, "Tiyuntsong: A self-play reinforcement learning approach for ABR streaming," in *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, New Orleans, LA, USA, 2021, pp. 1–14.

[16] Y. Zhang, S. Liu, Z. Yu, and Y. Sun, "LSTM-based rate adaptation for video streaming," *Multimedia Tools and Applications*, vol. 79, no. 15–16, pp. 10651–10666, 2020.

[17] T. Huang and Q. He, "A GRU-based QoE-driven adaptive bitrate streaming system," *Journal of Visual Communication and Image Representation*, vol. 74, art. no. 103000, 2020.

[18] M. Mino, Y. Kawahara, and H. Yasuda, "Energy-aware adaptive video streaming with QoE optimization for mobile clients," *IEEE Access*, vol. 8, pp. 18867–18880, 2020.

[19] S. Zhou, H. Yu, and M. Zeng, "Lightweight neural ABR for mobile edge video streaming," in *Proc. IEEE International Conference on Communications (ICC)*, Seoul, South Korea, 2022, pp. 1–6.

[20] S. Zarei and Y. Wang, "QoE-aware multipath forwarding in Named Data Networking," *Computer Communications*, vol. 149, pp. 193–204, 2020.

[21] J. Wu, Z. Du, and Y. Yu, "NDNLive: Live video streaming in Named Data Networking," in *Proc. IEEE INFOCOM Workshops*, Paris, France, 2019, pp. 1–6.

[22] J. Li, Z. Zhang, and M. Guo, "Machine learning-based interest forwarding in Named Data Networking," *IEEE Access*, vol. 8, pp. 132410–132422, 2020.

[23] Netflix. Internet connection speed recommendations. [Online]. Available: https://help.netflix.com/en/node/306

[24] H. N. Arifin, N. R. Syambas, and T. Juhana, "Implementation of a hybrid-based ABR algorithm in Named Data Networking video streaming systems," in *Proc. 10th International Conference on Wireless and Telematics*, Batam, Indonesia, 2024, pp. 1–5.

[25] Bilsyp. Ndn-public-master. GitHub repository. [Online]. Available: https://github.com/Bilsyp/ndn-public-master

[26] R. Netravali, A. Goyal, and H. Balakrishnan, "Mahimahi: Accurate record-and-replay for HTTP," in *Proc. USENIX NSDI*, Oakland, CA, USA, 2015, pp. 417–429.

[27] J. Woo, S. Hong, D. Kang, and D. An, "Improving the quality of experience of video streaming through a buffer-based adaptive bitrate algorithm and gated recurrent unit-based network bandwidth prediction," *Applied Sciences*, vol. 14, no. 22, 10490, 2024.