Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Trajectory-Optimized Offline Reinforcement Learning for Short-Video Adaptive Bitrate Streaming (ChinaMM 2026)

Chenyu Ji
Nanjing University
Bowei Xu
Nanjing University
Hao Chen
chenhao1210@nju.edu.cn
Nanjing University
Zhan Ma
Nanjing University

Research Article

Keywords: short video streaming, adaptive bitrate streaming, offline reinforcement learning, trajectory optimization, quality of experience

Posted Date: November 24th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11140068/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Trajectory-Optimized Offline Reinforcement Learning for Short-Video Adaptive Bitrate Streaming (ChinaMM 2026)

Chenyu Ji[1†], Bowei Xu[1†], Hao Chen[1*], Zhan Ma[1]

[1]School of Electronic Science and Engineering, Nanjing University, Xianlin Avenue, Nanjing, 210023, Jiangsu, China.

*Corresponding author(s). E-mail(s): chenhao1210@nju.edu.cn;
Contributing authors: nhjchy@126.com; xubowei@smail.nju.edu.cn; mazhan@nju.edu.cn;
†These authors contributed equally to this work.

Abstract

Purpose: Short-video adaptive bitrate (SABR) streaming requires jointly optimizing video prefetching and bitrate selection under dynamic network conditions and uncertain user swipe behavior. Existing heuristic-based methods often rely on manually designed rules, while online reinforcement learning (RL) incurs costly trial-and-error exploration. This work aims to develop an offline RL framework that improves streaming utility without requiring online exploration.

Methods: We propose OSCAR, a trajectory-optimized offline reinforcement learning framework for SABR. OSCAR first aggregates behavioral trajectories generated by multiple existing SABR algorithms and applies offline trajectory optimization to improve QoE-oriented bitrate decisions and suppress bandwidth-inefficient prefetching. A distributional Deep Q-Network with an LSTM-based temporal encoder is then used for value estimation, while Critic-Regularized Regression performs offline policy improvement by emphasizing advantageous actions supported by the logged data.

Results: Experiments on MMGC2022 and DUASVS video traces combined with OBOE/FCC network traces show that OSCAR consistently improves the overall utility compared with existing rule-based and learning-based SABR methods. In particular, OSCAR achieves an average utility improvement of 23.7% over the strongest baseline under the DUASVS+OBOE/FCC configuration. Ablation studies further demonstrate the complementary contributions of trajectory optimization, temporal modeling, and CRR-based policy improvement.

Conclusion: OSCAR provides an effective offline learning framework for short-video streaming that jointly improves playback quality and bandwidth efficiency while avoiding costly online exploration.

Keywords: short video streaming, adaptive bitrate streaming, offline reinforcement learning, trajectory optimization, quality of experience

1 Introduction

In recent years, the proliferation of short video platforms such as Douyin and Kuaishou has dramatically reshaped online video consumption, shifting user demand toward highly responsive and personalized experiences. Compared with traditional long-form video streaming services [1–3], short video services allow users to switch to the next video at any time by simply swiping the screen. As illustrated in Fig. 1, video chunks are encoded at multiple bitrates and stored on content delivery network (CDN) nodes. The client pulls these chunks on demand and maintains a local buffer for each video in the playback queue, which includes the currently playing video and several recommended videos. While prefetching recommended videos is essential to eliminate startup latency [4], it risks significant bandwidth waste if a user swipes before the downloaded content is viewed [5]. Consequently, Short-video Adaptive Bitrate (SABR) streaming presents a unique challenge: it requires jointly managing prefetching and bitrate selection across a list-wise queue, balancing the conflicting objectives of startup delay, visual quality, and bandwidth efficiency under fluctuating network conditions.

Current SABR approaches primarily fall into two categories: rule-based heuristics [6] and learning-based methods [7–9]. Rule-based algorithms, such as PDAS [10] from the ACM Multimedia 2022 Grand Challenge (MMGC2022), leverage handcrafted buffer management logic and model predictive control (MPC [2]) to optimize decisions. However, their fixed decision rules and sensitivity to hyperparameter tuning often limit generalization across diverse network patterns and user behaviors. To address these limitations, learning-based methods like DAM [11] and Incendio [12] utilize Reinforcement Learning (RL) to achieve stronger environmental adaptability. Nevertheless, their real-world deployment remains constrained by a formidable "sim-to-real" gap [13, 14]. Discrepancies in network dynamics and system implementation details, which are often oversimplified in simulators, cause policies that appear optimal in controlled settings to suffer severe performance degradation in live deployments. While online exploration could theoretically bridge this gap, the resulting "trial-and-error" process is prohibitively costly for production systems. As quantified in Fig. 2, the initial exploration phase can trigger a 65% spike in bandwidth waste and persistent QoE degradation. Such volatility is untenable for real-world platforms that require near-optimal performance and high bandwidth efficiency from day one.

Offline reinforcement learning has emerged as a promising paradigm that addresses these concerns by learning policies solely from static, pre-collected datasets without any online interaction with the environment [15]. By leveraging historical logs, offline RL is particularly suitable for industrial video streaming, where online data collection

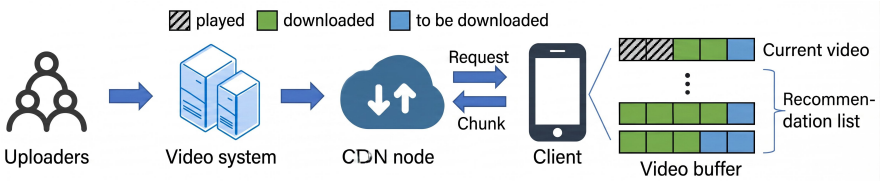

Fig. 1 Short video streaming system overview.

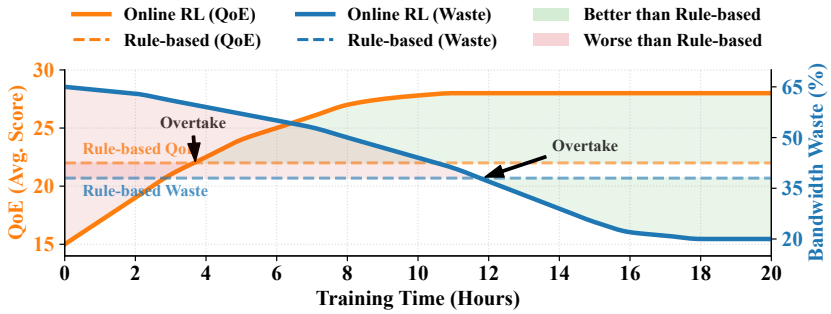

Fig. 2 Bandwidth waste and low QoE during online reinforcement learning.

is expensive or disruptive to user experience. Recent studies, such as Backwave [16], have demonstrated the potential of offline RL to narrow the sim-to-real gap in large-scale deployments.

However, applying offline RL to SABR introduces two fundamental challenges. The first is the distributional shift, i.e., the mismatch between the static logged dataset and the live deployment environment. Since offline RL algorithms cannot correct errors through online interaction, they may query out-of-distribution (OOD) actions during deployment that were poorly represented in the training data, leading to unreliable Q-value estimates and severe performance degradation. This challenge is particularly acute in SABR, where network conditions exhibit high variability and user behavior patterns can shift over time. The second challenge is the trajectory suboptimality: historical trajectory datasets inevitably contain suboptimal decisions from legacy algorithms, such as conservative bitrates or excessive prefetching. These "noise" trajectories can trap the learned policy in local optima, hindering its ability to surpass the original data-collection policy.

To systematically address these issues, we propose OSCAR, a trajectory-optimized offline RL algorithm tailored for short-video adaptive bitrate streaming. OSCAR first constructs a high-quality offline dataset by aggregating decision trajectories from multiple state-of-the-art SABR methods and distilling them to retain only the most high-QoE, bandwidth-efficient samples across diverse network conditions and user preferences. It then employs an LSTM-integrated Deep Q-Network (DQN) to encode the temporal dependencies in sequential streaming states. To counteract the distributional shift inherent in offline RL, we incorporate Critic-Regularized Regression (CRR), which stabilizes offline optimization by confining policy updates to state-action

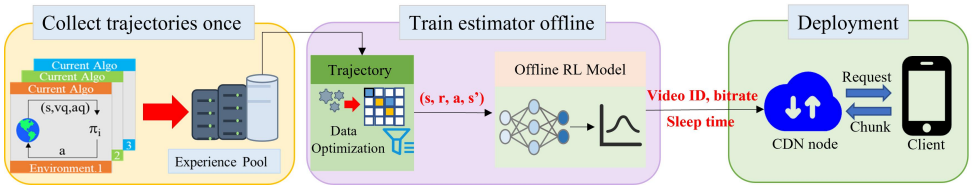

Fig. 3 OSCAR system overview.

pairs with positive advantage, thereby ensuring robust and safe learning from static data.

In general, we summarize our contributions as follows:

- We propose OSCAR, to the best of our knowledge, the first offline RL framework tailored specifically for SABR. It learns an optimal policy from historical trajectory data, while eliminating the costly trial-and-error exploration inherent in online RL.
- We introduce a dual-optimization strategy that combines trajectory optimization for improving the quality of offline behavioral data with critic-regularized regression for mitigating distributional shift during offline policy learning. This design enables OSCAR to learn an effective SABR policy without online trial-and-error exploration.
- Extensive experiments using real-world video and network traces demonstrate that OSCAR outperforms state-of-the-art SABR algorithms, improving the average utility score by 23.7% while maintaining superior generalization across diverse environments.

2 Design

2.1 OSCAR Overview

Unlike generic offline RL approaches that passively utilize collected data, OSCAR is a trajectory-optimized offline reinforcement learning-based algorithm tailored specifically for SABR with list-wise prefetching. The overall workflow of OSCAR consists of three key stages, as depicted in Fig. 3. First, OSCAR aggregates decision trajectories from multiple advanced SABR methods under diverse network conditions and user preference scenarios, constructing a comprehensive offline dataset composed of state-action-reward tuples. Second, a dedicated trajectory optimization module is designed to refine these raw trajectories, with dual objectives of maximizing user QoE and minimizing bandwidth waste. This optimization process enhances the quality of the training corpus, thereby improving its suitability for policy stitching and enabling more effective offline policy learning— a capability critical for surpassing the performance of individual data collection strategies. Leveraging the optimized dataset, OSCAR adopts a DQN-based offline RL framework, augmented with domain-specific components to address the inherent challenges of learning from static offline data. Specifically, the framework integrates an LSTM network to capture temporal dependencies in sequential streaming states, which is essential for modeling dynamic user behavior and network fluctuations. Additionally, it incorporates CRR to mitigate

distributional shift—a core challenge in offline RL—by restricting policy updates to state-action pairs with positive advantage, effectively filtering out inferior actions and focusing learning on beneficial decisions. Finally, the trained OSCAR policy is evaluated in the trace-driven short-video streaming environment to execute joint prefetching and bitrate decisions under diverse network and user-behavior conditions.

We employ a distributional DQN [17] as the critic backbone of OSCAR. This design is well suited to the SABR problem because the controller operates over a compact 16-action discrete action space and requires accurate estimation of long-term streaming returns. Unlike standard DQN, which represents each action value by a single expectation, the distributional formulation models the return distribution over a set of discrete supports, providing richer value estimates for heterogeneous streaming trajectories involving different network conditions, buffer states, and user behaviors.

Importantly, DQN does not constitute the complete offline policy-learning mechanism in OSCAR. Instead, OSCAR separates value estimation from offline policy improvement. The distributional DQN serves as the critic for estimating return distributions and corresponding action values, while Critic-Regularized Regression (CRR) performs offline policy improvement through advantage-weighted behavioral regression. In particular, CRR constrains policy updates toward advantageous actions observed in the offline dataset, thereby reducing the influence of unsupported or inferior actions during policy optimization. We denote this combination as DQN+CRR throughout the evaluation.

To examine whether this design is competitive with alternative offline learning formulations rather than relying only on the choice of DQN, Section 3.3 further compares DQN+CRR with Behavior Cloning (BC), offline DQN, Discrete Implicit Q-Learning (IQL), and Discrete Conservative Q-Learning (CQL), with all methods trained on the same original raw offline trajectories. The contribution of trajectory optimization is then evaluated separately through the stepwise component ablation in Section 3.6.1.

2.2 Offline Dataset Construction via Behavioral Trajectories

Unlike online reinforcement learning that requires costly real-world interactions and exploration trials, offline RL enables direct policy learning from pre-collected behavioral trajectories. High-quality trajectories are critical for offline RL, as they provide diverse decision samples and accurately capture essential system dynamics of the target task. To construct such a high-quality dataset for SABR, we aggregate trajectories from state-of-the-art SABR algorithms under a wide range of network conditions and user preference scenarios. Each sample in our dataset is formatted as a (*state*, *action*, *reward*) tuple, with the definition of each component provided below.

State definition: At each decision step t, the state s_t is defined as $s_t = (\vec{b_t}, \vec{l_j}, \vec{g_j}, \vec{u_j}, \vec{h_j}, \vec{q_j}, \vec{f_j})$. Here, $\vec{b_t}$ is the throughput vector observed over the past k video chunks (i.e., $\vec{b_t} = \{b_{t-k+1}, \ldots, b_t\}$), and the throughput of each chunk is calculated by dividing the chunk size by the download duration. In this work, we set $k = 5$ to effectively extract short-term network dynamics while maintaining computational feasibility. The remaining states represent different types of factors for each video $j \in [1, 5]$ in the playing queue. Specifically, $\vec{l_j}$ represents the conditional probability that a user will continue watching from the current chunk m to a future chunk n in

video j. It is calculated as:

$$l_j = \frac{p_j^n}{p_j^m}, \tag{1}$$

where p_j^m and p_j^n are the statistically derived user-retention rates of chunks m and n, respectively, for video j.

In a practical streaming system, these retention rates can be estimated from historical viewing logs. Let $N_j(m)$ denote the number of viewing sessions that reach chunk m of video j, and let $N_j(1)$ denote the number of sessions that start watching the video. The empirical retention rate at chunk m can then be estimated as $p_j^m = N_j(m)/N_j(1)$. Accordingly, given that a user has already reached chunk m, the conditional probability of continuing to chunk n can be equivalently estimated as $l_j = N_j(n)/N_j(m)$. In our trace-driven evaluation, we directly use the retention statistics provided by the video dataset.

It is worth noting that the retention probability is a statistical estimate rather than an exact observation. Estimation errors can therefore affect OSCAR's decisions. An overestimated retention probability assigns greater importance to the quality of future chunks in the reward function and may encourage more aggressive high-bitrate downloading or prefetching, potentially increasing bandwidth wastage when the user swipes away earlier than expected. Conversely, an underestimated retention probability may make the policy more conservative, reducing unnecessary prefetching but potentially sacrificing achievable QoE. Retention estimates are also used in the retention-aware adaptive buffer threshold introduced in Section 2.3, so estimation errors may further affect the trade-off between playback quality and bandwidth efficiency.

g_j represents the current buffer size of video j; u_j is the average size of the next video chunk of video j at different bit rates; h_j is the rebuffering time caused by downloading the last chunk of video j, with a value of 0 if there is no rebuffering; q_j represents the bitrate at which the last chunk of video j was downloaded; and f_j measures the abruptness of quality changes, calculated as the absolute difference between consecutive chunk bitrates:

$$f_j = |q_j - q_{j-1}|. \tag{2}$$

Action space design: In the context of short video streaming, the agent is required to make three interrelated decisions: the target video chunk to prefetch, the download bitrate, and whether to pause prefetching. The pause action is distinct from the other two, as selecting it means no video chunks will be downloaded. We thus design a discrete action space consisting of 16 actions: the first 15 correspond to combinations of video chunks and download bitrates, numbered from (video0, level0) to (video4, level2), where the three bitrate levels map to $\{750,\ 1200,\ 1800\}$ kbps respectively. The 16th action is a pause operation with a fixed duration $\tau = 250$ms, which balances optimal computing resource utilization and avoidance of missing ideal decision timings.

Reward function formulation: The reward function serves as the guiding signal for the offline RL agent, aligning its learning objectives with the dual goals of SABR: maximizing QoE while minimizing unnecessary bandwidth waste. The reward at step

t is formulated as:

$$r_t = w_n \cdot l_n \cdot (R_n - S_n) - \mu \cdot T_n - \nu \cdot bw_n, \tag{3}$$

where R_n and S_n represent the bitrate and its fluctuation of the played chunk n, promoting high playback quality while discouraging quality variations; T_n and bw_n denote the rebuffering duration and wasted bandwidth caused by downloading chunk n, with corresponding penalty weights μ and ν. To tailor the reward function to the unique dynamics of short-video streaming (e.g., frequent user swipes and ephemeral viewing patterns), we introduce two key adaptations:

- Conditional-watch probability l_n: the likelihood that the user will continue watching the current chunk. By scaling the quality term (R_n - S_n) with l_n, the agent's incentive to prefetch is reduced when the user is likely to swipe away, encouraging pause decisions and bandwidth conservation.
- Bandwidth-aware indicator w_n: a binary variable that denotes whether the downloaded chunk n is actually viewed by the user, formally defined as:

$$w_n = \begin{cases} 0, & \text{if chunk } n \text{ is not viewed by the user} \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

  This term nullifies the quality reward for unwatched chunks, explicitly penalizing wasteful prefetching behavior and aligning the agent's decisions with actual resource utilization.

Collectively, these adaptations enable the reward function to jointly capture perceptual quality, playback smoothness, and bandwidth efficiency, guiding the learned policy toward high-QoE decisions with minimal data waste.

To characterize the empirical coverage of the collected offline dataset, we analyze the distributions of several key state features and actions. As illustrated in Fig. 4, the collected trajectories span a broad range of observed throughput, buffer occupancy, and user-retention probabilities. In addition, all 16 discrete actions appear in the training dataset with nonzero frequency.

These results provide an empirical characterization of the state features and actions observed under the evaluated training configurations. However, because the SABR state space is high-dimensional, the marginal feature and action distributions alone do not imply complete joint state–action coverage or guarantee the absence of out-of-distribution states. We therefore use Fig. 4 as a quantitative description of the collected offline data rather than as a guarantee of coverage under arbitrary deployment conditions.

2.3 Trajectory Optimization for Enhanced Offline Policy Stitching

A core promise of offline reinforcement learning lies in stitching sub-trajectories from diverse behavioral patterns to synthesize a policy that outperforms the original strategies. By integrating good decisions across different trajectories, the learner can discover

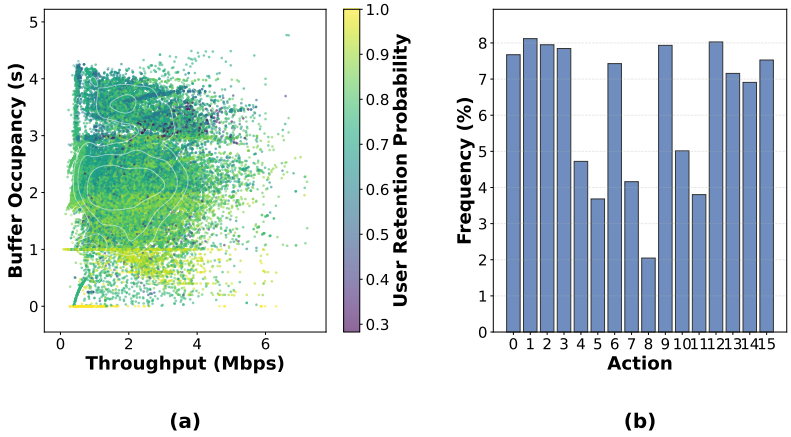

Fig. 4 Training dataset coverage analysis: (a) distribution of throughput, buffer occupancy, and user retention probability; (b) action frequency across the 16 discrete actions.

optimal strategies that were never fully executed in any single logged session. However, even trajectories collected from state-of-the-art algorithms inevitably contain suboptimal decisions—such as unnecessarily low bitrates when bandwidth and buffer conditions safely support higher quality, or wasteful prefetching of chunks unlikely to be watched—that disrupt decision chains and hinder effective policy stitching.

To address this limitation, we introduce a dual-objective trajectory optimization module to curate the offline dataset before policy training. The module is designed as an offline data-quality enhancement procedure rather than an online rule-based controller. It operates through two complementary mechanisms: repairing conservative bitrate decisions when a higher-bitrate alternative remains feasible under the logged trajectory, and suppressing excessive prefetching decisions according to a retention-aware adaptive buffer threshold.

Importantly, these rules are applied exclusively to historical trajectories before policy training and are never used to override the learned policy during inference. The QoE-oriented procedure uses zero-valued feasibility boundaries derived from buffer and temporal constraints, whereas the bandwidth-oriented procedure contains a configurable conservative buffer threshold. Their respective roles and sensitivity are discussed below.

QoE-oriented optimization via repairing conservative breaks: To improve user QoE, we focus on conservative decisions in the logged trajectories, where a relatively low bitrate is selected even though a higher-bitrate chunk could have been downloaded without causing rebuffering or affecting the subsequent logged decision.

To identify such decisions, let B_k^{logs} denote the logged buffer occupancy at decision step k, and let d_k^{new} denote the estimated download time when the current chunk is replaced by a higher-bitrate version. We define the post-download buffer B_k^{after} and the temporal slack T_{slack} as

$$
\begin{aligned}
B_k^{\text{after}} &= B_k^{\text{logs}} - d_k^{\text{new}}, \\
T_{\text{slack}} &= B_k^{\text{logs}} + L - d_k^{\text{new}} - B_{k+1}^{\text{logs}},
\end{aligned}
\tag{5}
$$

where L is the chunk duration and B_{k+1}^{logs} denotes the logged buffer occupancy at the next decision step.

A logged bitrate decision is regarded as a feasible bitrate-upgrade candidate only when both

$$
B_k^{\text{after}} > 0, \qquad T_{\text{slack}} > 0 \tag{6}
$$

are satisfied. Equivalently, we define the QoE-oriented trajectory-optimization indicator as

$$
\chi_k^{\text{QoE}} = \mathbb{I}\left[B_k^{\text{after}} > 0\right] \mathbb{I}\left[T_{\text{slack}} > 0\right], \tag{7}
$$

where $\mathbb{I}[\cdot]$ denotes the indicator function. Only when $\chi_k^{\text{QoE}} = 1$ is the original lower-bitrate decision identified as an overly conservative decision and replaced by the feasible higher-bitrate alternative. Otherwise, the original logged decision is retained unchanged.

Importantly, the zero boundaries in the above conditions are not empirically tuned heuristic thresholds. Instead, they directly represent feasibility conditions of the logged streaming trajectory. Specifically, $B_k^{\text{after}} > 0$ ensures that downloading the higher-bitrate chunk does not exhaust the available playback buffer, while $T_{\text{slack}} > 0$ ensures that sufficient temporal slack remains with respect to the logged buffer state at the subsequent decision. Therefore, the trajectory optimization modifies a bitrate decision only when the higher-bitrate alternative remains feasible under both conditions.

We note that decisions arbitrarily close to these feasibility boundaries may still be sensitive to estimation errors; therefore, the above conditions should be interpreted as trajectory-based feasibility tests rather than as a guarantee that all boundary cases are error-free.

Bandwidth-oriented optimization via adaptive buffer control: Although prefetching future videos can reduce startup delay, excessive prefetching may waste substantial bandwidth because users can swipe away before the downloaded content is viewed. To identify bandwidth-inefficient decisions in the logged trajectories, we define an adaptive maximum buffer threshold for chunk m of video i as

$$
b_{i,m}^{\max} = \max\left\{p_{i,m}(m_c) \cdot T_{i,m}^{\max},\ b_i^{th}\right\}, \tag{8}
$$

where $p_{i,m}(m_c)$ denotes the conditional probability that the user will watch chunk m given the current playback progress m_c, $T_{i,m}^{\max}$ denotes the download time of chunk m at the highest available bitrate, and b_i^{th} is a conservative lower bound on the buffer threshold. In our implementation, b_i^{th} is set to the chunk duration L.

For a logged prefetching decision, if the buffer occupancy of the corresponding video already exceeds its adaptive threshold,

$$
B_{i,m}^{\text{logs}} > b_{i,m}^{\max}, \tag{9}
$$

but the logged policy still initiates an additional prefetching operation; we regard this decision as bandwidth inefficient and remove it during offline trajectory curation. Otherwise, the logged decision is retained.

It is important to clarify that the adaptive buffer threshold is used only as a hard criterion during offline trajectory curation. It is not included as a soft penalty term in the reward function and is not enforced as an online action constraint during deployment. After the optimized offline dataset is constructed, the learned OSCAR policy makes its decisions directly from the observed streaming state without applying this buffer rule during inference.

Unlike the zero-valued feasibility boundaries used in the QoE-oriented optimization, b_i^{th} is a configurable conservative threshold. Its value determines how aggressively the curation procedure suppresses future-video prefetching. We therefore evaluate its sensitivity in Section 3.6.2 by varying b_i^{th} across different values and examining the resulting QoE–bandwidth trade-off.

Overall, the two trajectory-optimization mechanisms serve distinct but complementary purposes. The QoE-oriented mechanism identifies feasible opportunities to replace overly conservative bitrate decisions, whereas the bandwidth-oriented mechanism suppresses prefetching decisions that exceed the retention-aware adaptive buffer threshold. Both mechanisms operate only on historical trajectories before policy training, and neither introduces an additional rule-based control stage during OSCAR inference.

Although the trajectory-optimization module is applied offline, it still incorporates domain knowledge into the data-curation process and therefore cannot completely eliminate selection bias. In particular, decisions close to the QoE feasibility boundaries may be sensitive to estimation errors, while the bandwidth-oriented optimization depends on the configurable threshold b_i^{th}. The sensitivity of this threshold is explicitly examined in our evaluation. More general approaches, such as constrained offline reinforcement learning or Lagrangian formulations that directly incorporate buffer and bandwidth constraints into the policy-learning objective, provide an alternative direction to rule-based offline data curation.

2.4 Learning with Temporal Features and Regularization

To improve offline policy learning for SABR, we augment the distributional DQN framework with two complementary components: an LSTM-based temporal encoder and Critic-Regularized Regression (CRR). The LSTM encoder captures temporal dependencies across sequential streaming states, including multi-step throughput trends, the evolution of multiple video buffers caused by previous prefetching decisions, and changes in user-watch probability as playback progresses. By recursively aggregating these historical signals, the LSTM provides a temporally informed representation for subsequent value estimation and policy learning.

CRR is incorporated as an offline policy-improvement mechanism. Rather than modifying or removing samples from the offline dataset, CRR assigns different contributions to logged state–action pairs according to their estimated advantages. Actions with positive estimated advantage are reinforced by the policy objective, whereas actions with non-positive advantage receive zero weight in the policy-improvement

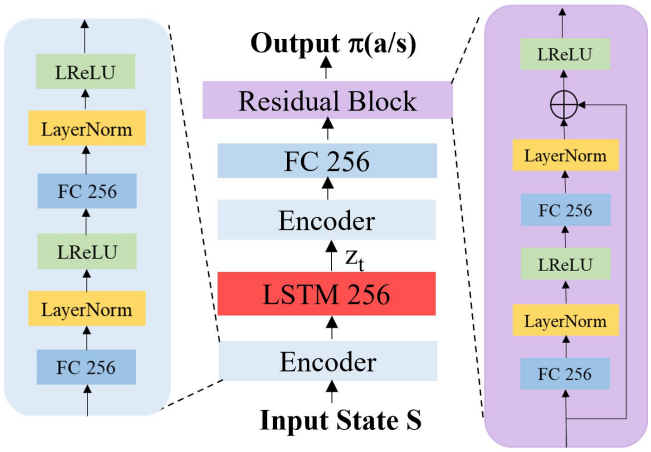

Fig. 5 OSCAR's neural network.

loss. In this way, the distributional DQN serves as the critic for value estimation, while CRR constrains policy improvement toward advantageous actions supported by the behavioral data. The detailed network architecture and CRR objective are introduced below.

Network architecture: To effectively capture the sequential nature of short-video streaming and mitigate distributional shift in offline learning, we design a neural network architecture that integrates an LSTM network into a distributional Deep Q-Network. As illustrated in Fig. 5, the raw sequential state s_t, which includes throughput history, buffer states, and user-watch probabilities, is first processed by a two-layer LSTM network. This layer captures the long-range dependencies inherent in user swipe behavior, network throughput fluctuations, and multi-buffer dynamics. The LSTM's gating mechanisms dynamically filter historical information, and its hidden state effectively summarizes the cumulative influence of past observations, producing a temporally-aware feature vector z_t.

This feature vector z_t is then passed to the distributional head of the DQN. Instead of outputting a scalar Q-value, the head models a full probability distribution $p(s, a)$ over a discrete set of value supports z_i (where $i = 0, 1, ..., N-1$ and N is the number of atoms). This distributional formulation explicitly represents the uncertainty and multimodality of future returns, providing more robust learning signals when confronting noisy or inconsistent samples in the offline dataset. To stabilize the training of this distributional value function, we further adopt a double DQN scheme: the online network uses z_t to choose the next action a^*, while a target network evaluates the corresponding value distribution $p(s', a^*)$. Given the immediate reward r, discount factor γ, support bounds $v_{\min}$ and $v_{\max}$, and support spacing $\Delta z = (v_{\max} - v_{\min})/(N-1)$, the distributional TD target is constructed by projecting $r + \gamma z_j$ onto the discrete support set. Each projected target is first clipped to the support range:

$$\mathcal{T} z_j = \min\left(\max(r + \gamma z_j,\ v_{\min}),\ v_{\max}\right), \tag{10}$$

and then proportionally distributed to its two nearest support atoms via the projection:

$$m_i = \sum_{j=0}^{N-1} \left[1 - \frac{|\mathcal{T} z_j - z_i|}{\Delta z}\right]_0^1 \cdot p_j(s', a^*), \tag{11}$$

where $[\cdot]_0^1$ denotes clipping the result to the interval $[0, 1]$. The cross-entropy loss L_{TD} between the predicted distribution $p(s, a)$ and the target distribution m is minimized:

$$L_{\text{TD}} = -\sum_{i=0}^{N-1} m_i \log p_i(s, a). \tag{12}$$

Policy improvement: The short-video streaming environment poses a distinct challenge for offline RL: the static logged data often mismatches the dynamic real-world settings, causing a naively trained policy to overfit to sub-optimal decisions or over-extrapolate on unseen states. To address this issue and learn a safe yet high-performing policy, we employ CRR to train a separate policy network $\pi_\phi(a|s)$. CRR operates by weighting state-action pairs according to their estimated advantage, thereby focusing learning on high-quality decisions. We first compute the state-value function $V(s)$ as the average Q-value over all actions, and the advantage function $A(s, a)$ to quantify how much better a specific action a is compared to the average action at state s:

$$V(s) \approx \frac{1}{|A|} \sum_a Q(s, a), \quad A(s, a) = Q(s, a) - V(s). \tag{13}$$

To filter out inferior state-action pairs, we adopt a binary weighting scheme: $w(s, a) = \mathbb{I}[A(s, a) > 0]$, where $\mathbb{I}[\cdot]$ is the indicator function. This scheme restricts policy improvement to logged actions with positive estimated advantage. Logged actions with non-positive advantage are assigned zero weight in the policy-improvement loss, but their transitions remain in the offline dataset and can still be used for critic/value learning. The policy network is then trained to maximize the likelihood of these advantaged actions via the advantage-weighted negative log-likelihood loss:

$$L_\pi = -\mathbb{E}_{(s,a)\sim D}[w(s, a) \cdot \log \pi_\phi(a|s)], \tag{14}$$

where D denotes the offline dataset. By employing advantage-weighted maximum likelihood, CRR restricts the policy to actions supported by the offline data, reinforcing high-QoE, low-bandwidth-waste decision patterns and further mitigating distributional shift.

Training and inference: The total training loss of the augmented DQN framework is the sum of the distributional value learning loss (L_{TD}) and the policy improvement loss (L_π):

$$L_{\text{total}} = L_{\text{TD}} + L_\pi. \tag{15}$$

We optimize the model end-to-end using the Adam optimizer, combined with cosine annealing for learning rate scheduling and gradient clipping to prevent training

instability. Target networks are softly updated to enhance training stability. During inference, the agent extracts temporal features from the state sequence via the LSTM network and samples actions directly from the trained policy network, enabling real-time decision-making tailored to dynamic SABR scenarios.

3 Evaluation

3.1 Methodology

Short-video streaming simulator: To the best of our knowledge, no open-source real-world platform currently enables fair and comprehensive comparison of algorithms for the list-wise prefetching SABR scenario. We thus adopt the multi-video simulator from MMGC2022, which serves as the de facto benchmark in this field and is widely utilized by related works [10–12, 18]. The simulator generates diverse streaming sessions by randomly pairing video traces with network traces. Each video trace includes chunk sizes across multiple bitrates and a user-retention trace that quantifies the probability of continuing to watch each chunk.

Network and video traces: To maximize realism within this controlled setting, we construct a comprehensive dataset by integrating real-world data: public network datasets (OBOE [3] and FCC [19]) and video traces from DUASVS [20]. This integration captures authentic dynamics of bandwidth variation and user behavior to the greatest extent possible. The dataset is split into 80% for training and 20% for testing. All compared algorithms are evaluated on the same held-out test set to ensure a fair comparison. Notably, the training paradigms differ across methods: OSCAR is trained offline on decision trajectories generated by all five SABR baselines used in our end-to-end evaluation: Mpc+fixed-preload, PDAS, DHP, DAM, and Incendio. These trajectories are collected exclusively from the training portion of the video–network pairs. The same five methods are also used as evaluation baselines, but all reported performance comparisons are conducted on the common held-out test set. Therefore, no trajectory from the held-out test video–network pairs is used for OSCAR training. This setup ensures that OSCAR avoids costly online exploration while being evaluated under identical test conditions.

SABR algorithms: Recent SABR works have primarily focused on streaming-oriented scenarios, where bitrate adaptation is performed during the playback of a single video (closely resembling traditional long-form video ABR). These algorithms are not directly applicable to our problem, as they fail to address the joint prefetching and playback decisions across multiple videos in a queue. In contrast, our work targets the list-wise prefetching scenario—a core mechanism that distinguishes short-video platforms from traditional video services, directly impacting user swipe experience and bandwidth efficiency. To ensure fair and meaningful comparisons, we select baselines that are specifically designed for or adaptable to the list-wise prefetching setting:

- Mpc+fixed-preload: Combines the prevalent model-based ABR approach (Robust Model Predictive Control, RobustMPC) with the sleeping mechanism from fixed-preload [21], serving as the official baseline of MMGC2022. RobustMPC maximizes

the accumulated utility score over a 5-chunk horizon based on buffer occupancy observations and throughput predictions.

- PDAS [10]: A high-performance rule-based algorithm that synergizes a user retention model with hand-crafted buffer heuristics. It employs RobustMPC to simulate future states and optimize prefetching/bitrate decisions under uncertainty. As the state-of-the-art method from MMGC2022, we implement PDAS faithfully following its original paper.
- DHP [18]: A heuristic algorithm for short video streaming that integrates a retention-aware maximum buffer model with a dynamic parameter adaptation mechanism. The coefficients of its QoE objective are tuned via sigmoid, exponential, and inverse functions based on observed bandwidth. We implement DHP faithfully following its original paper.
- DAM [11]: A deep reinforcement learning-based approach that trains a control policy to jointly decide video ID, bitrate level, and pause/sleep time using an action masking mechanism. It ranked first among all learning-based techniques in MMGC2022. We faithfully implement DAM following its original paper.
- Incendio [12] A state-of-the-art multi-agent RL approach that employs imitation learning for pre-training and RL for fine-tuning, balancing training efficiency and final performance. We faithfully implement Incendio following its original paper.

Offline RL baselines and experimental protocol: To separately evaluate the offline policy-learning mechanism, we compare DQN+CRR with four representative offline learning baselines: Behavior Cloning (BC), offline DQN, Discrete Conservative Q-Learning (CQL), and Discrete Implicit Q-Learning (IQL).

BC directly learns the behavioral policy from logged state–action pairs and therefore serves as a non-value-based offline learning baseline. Offline DQN directly learns action values from the fixed dataset without introducing an offline-specific policy-improvement constraint. Discrete CQL applies conservative value regularization to suppress overestimated values of actions insufficiently supported by the offline data, while Discrete IQL performs policy improvement primarily from high-value in-distribution actions without explicitly maximizing over out-of-distribution actions.

For this comparison, BC, offline DQN, Discrete IQL, Discrete CQL, and DQN+CRR are all trained on exactly the same original raw offline trajectories. They use the same state representation, 16-action discrete action space, temporal encoder, training/test split, training budget, and held-out evaluation sessions. No trajectory optimization is applied in this experiment. Therefore, the comparison in Section 3.3 isolates the effect of the offline policy-learning formulation. The contribution of trajectory optimization is evaluated separately through the stepwise component ablation in Section 3.6.1.

All reported performance values are computed over the same held-out set of video–network sessions. Unless otherwise specified, the tables report the average performance across these held-out evaluation sessions.

Evaluation metric: The primary evaluation metric is the comprehensive utility score U_i, which jointly quantifies video quality and bandwidth efficiency. It is defined

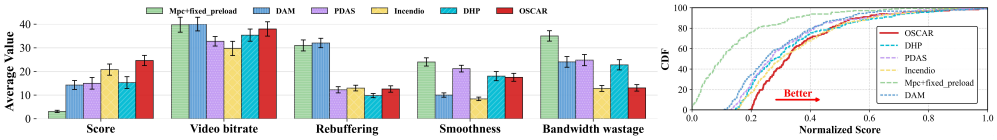

Fig. 6 Comparing OSCAR with existing schemes in terms of the average performance and full CDF performance on MMGC2022 video and bandwidth traces.

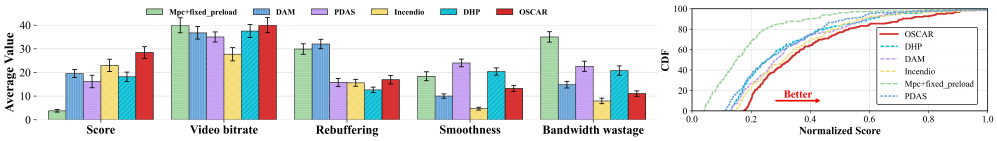

Fig. 7 Comparing OSCAR with existing schemes in terms of the average performance and full CDF performance on DUASVS video traces and Oboe/FCC bandwidth traces.

as:

$$\begin{aligned} U_i &= QoE_i - Wastage_i \\ &= \sum_m (R_m - S_m) - \sum_n \mu \cdot T_n - \sum_n \nu \cdot bw_n, \end{aligned} \tag{16}$$

where R_m and S_m denote the bitrate and bitrate variation of each played chunk, capturing perceptual quality and playback smoothness; T_n and bw_n represent rebuffering time and wasted bandwidth caused by downloaded chunks, respectively. Following the MMGC2022 benchmark, the penalty coefficients are set to $\mu = 1.85$ and $\nu = 0.5$.

3.2 Overall Results

Performance comparisons under different dataset configurations are presented in Figs. 6 and 7. As shown in the bar plots, OSCAR consistently achieves the highest average utility score across both test setups, outperforming the strongest baseline Incendio by 18.6% under the MMGC2022 configuration and by 23.7% under the DUASVS+OBOE/FCC configuration. The error bars indicate one standard deviation across the held-out video–network sessions. Under the MMGC2022 configuration, OSCAR achieves a mean utility score of 24.3 ± 2.3, compared with 20.5 ± 2.4 for Incendio. Under the DUASVS+OBOE/FCC configuration, the corresponding results are 28.41 ± 2.6 for OSCAR and 23.0 ± 2.6 for Incendio. These results show that OSCAR maintains a higher average utility across both evaluation settings while exhibiting a comparable level of session-level variation. The corresponding cumulative distribution function (CDF) curves further validate OSCAR's superior generalization ability: its curve lies farthest to the right in both scenarios, confirming that OSCAR delivers higher utility not only on average but also across the vast majority of individual streaming sessions. This consistent performance gain over diverse video and network traces underscores OSCAR's strong robustness to varying user behaviors and channel dynamics.

A detailed breakdown of performance metrics reveals the source of OSCAR's advantage across all key dimensions. Compared to traditional rule-based baselines (Mpc+fixed-preload, PDAS, DHP) and learning-based baselines (DAM, Incendio), OSCAR achieves a far more balanced trade-off: it delivers higher average video bitrates than the overly conservative Incendio, maintains smoother playback than the rigid DAM and resource-inefficient Mpc+fixed-preload, and incurs significantly lower bandwidth wastage than all competing schemes. This balanced optimization directly contributes to its superior overall utility, addressing the inherent limitations of both traditional rule-based and existing learning-based methods.

Further analysis demonstrates that rule-based methods suffer from severe generalization gaps. While PDAS and DHP achieve competitive performance on the MMGC2022 dataset, their effectiveness degrades sharply on the DUASVS+OBOE/FCC traces, where both are outperformed by the learning-based method DAM. This performance decline exposes the inherent limitations of handcrafted heuristics: PDAS's probability-driven buffer control and DHP's dynamic parameter adaptation are both tuned to the specific characteristics of the MMGC2022 dataset. When confronted with unseen network dynamics and user preference variations from independent data sources, these fixed or narrowly adaptive rules fail to generalize, whereas learning-based methods retain more robust performance. Notably, even among learning-based methods, there exist clear performance disparities. Incendio, despite its multi-agent RL design, adopts an overly conservative decision policy that prioritizes bandwidth efficiency at the cost of visual quality, resulting in the lowest average bitrate and overall utility among learning-based baselines. DAM, while outperforming rule-based methods on unseen data, exhibits inflexible bitrate preferences that struggle to adapt to extreme network conditions, leading to suboptimal smoothness and rebuffering performance. In contrast, OSCAR's context-aware, trajectory-optimized offline RL framework overcomes these limitations, achieving consistent superiority across both datasets and all evaluation metrics.

3.3 Comparison with Standard Offline RL Algorithms

To evaluate the effectiveness of OSCAR's offline policy-improvement mechanism independently of trajectory optimization, we compare our DQN+CRR formulation with several representative offline learning algorithms using the same original raw offline trajectories. Specifically, we consider Behavior Cloning (BC), offline DQN, Discrete Implicit Q-Learning (IQL), Discrete Conservative Q-Learning (CQL), and DQN+CRR.

BC directly imitates the logged actions and therefore serves as a non-value-based offline learning baseline. Offline DQN directly learns the action-value function from the fixed dataset without introducing an offline-specific policy-improvement constraint. Discrete CQL addresses distributional shift by conservatively penalizing Q-values associated with actions that are insufficiently supported by the offline data. Discrete IQL avoids explicitly maximizing over potentially out-of-distribution actions and instead performs policy improvement primarily using high-value actions supported by the dataset. In DQN+CRR, the distributional DQN serves as the critic for return

Table 1 Comparison with representative offline learning methods trained on the same raw offline trajectories.

Method	Average Utility Score	Average QoE	Average Bandwidth Wastage
BC	24.31	32.91	27.6%
Offline DQN	24.67	33.25	28.37%
Discrete IQL	25.08	33.83	26.8%
Discrete CQL	25.21	33.96	26.5%
DQN+CRR	**25.48**	**34.24**	**26.1%**

estimation, while CRR acts as the offline policy-improvement component through advantage-weighted behavioral regression.

For a fair comparison, all methods are trained on exactly the same unmodified offline trajectories and use the same state representation, 16-action discrete action space, temporal encoder, training/test split, training budget, and held-out evaluation sessions. No trajectory optimization is applied in this experiment. Therefore, this comparison isolates the effect of the offline policy-learning formulation from that of OSCAR's trajectory optimization module. The contribution of trajectory optimization is evaluated separately through the stepwise ablation study in Table 3.

As shown in Table 1, BC achieves an average utility score of 24.31, with an average QoE of 32.91 and bandwidth wastage of 27.6%. Although behavior cloning avoids explicit value extrapolation by directly imitating the logged behavior, its imitation objective limits its ability to preferentially improve upon the quality of the decisions contained in the offline dataset.

Offline DQN improves the average utility to 24.67 and the average QoE to 33.25. However, its bandwidth wastage increases to 28.37%. Because DQN performs value-based policy learning directly on the static behavioral dataset without an offline-specific policy constraint, inaccurate value estimates for insufficiently supported actions can affect policy improvement. This result illustrates the difficulty of directly applying conventional value-based reinforcement learning to a fixed offline dataset.

The offline-specific methods achieve more balanced performance. Discrete IQL obtains an average utility score of 25.08, an average QoE of 33.83, and bandwidth wastage of 26.8%. Discrete CQL further improves these metrics to 25.21, 33.96, and 26.5%, respectively. Compared with vanilla offline DQN, both methods improve overall utility while reducing unnecessary bandwidth consumption, indicating the benefit of explicitly accounting for distributional shift in offline SABR learning.

Among all evaluated formulations, DQN+CRR achieves the best overall performance, with an average utility score of 25.48, an average QoE of 34.24, and bandwidth wastage of 26.1%. Compared with offline DQN, DQN+CRR improves the average utility by approximately 3.3% and reduces bandwidth wastage from 28.37% to 26.1%. It also improves the average utility over Discrete IQL and Discrete CQL by approximately 1.6% and 1.1%, respectively. Since all methods are trained on the same raw trajectories and no trajectory optimization is involved in this comparison, these gains are attributable to the offline policy-learning mechanism rather than to data curation.

The improvement of DQN+CRR stems from the CRR policy-improvement mechanism. Instead of treating all logged state–action pairs equally during policy optimization, CRR uses the estimated advantage to determine their contribution to the policy-improvement objective. In OSCAR, we employ a binary weighting strategy defined as

$$w(s,a) = \mathbb{I}\left[A(s,a) > 0\right], \tag{17}$$

where $A(s,a)$ denotes the estimated advantage of action a in state s. The corresponding policy-improvement objective is

$$\mathcal{L}_{\mathrm{CRR}} = -\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[w(s,a)\log\pi_\theta(a|s)\right]. \tag{18}$$

Under this formulation, logged actions with positive estimated advantage contribute to policy improvement, whereas actions with non-positive advantage receive zero weight. This keeps the learned policy anchored to actions observed in the offline dataset while preferentially reinforcing higher-quality behavioral decisions. Consequently, CRR reduces the influence of inferior logged actions without requiring the policy to optimize over unsupported actions outside the behavioral distribution.

It is worth clarifying that the distributional DQN and CRR play different roles in our implementation. The distributional DQN serves as the critic backbone and estimates the return distribution and corresponding action values, whereas CRR provides the offline policy-improvement mechanism through advantage-weighted behavioral regression. We therefore denote this configuration as DQN+CRR throughout the evaluation. The notation emphasizes that CRR is integrated with the DQN-based critic rather than replacing the value estimation component.

We treat DQN, CQL, IQL, and DQN+CRR as alternative offline policy-learning formulations in this comparison rather than as arbitrarily composable modules. In particular, IQL already includes its own advantage-weighted policy-extraction mechanism, and combining it with CRR would replace or duplicate part of its standard policy improvement procedure. Similarly, although a conservative CQL critic could in principle be combined with a CRR-based policy-improvement objective, such a combination would constitute an additional hybrid algorithm rather than the standard CQL baseline. We therefore compare the standard formulations of IQL and CQL against our DQN+CRR configuration under the same raw-data setting.

Finally, this experiment deliberately excludes OSCAR's trajectory optimization module in order to isolate the contribution of the offline policy-learning mechanism. The complementary contribution of trajectory optimization is examined in Table 3, where CRR is first integrated with the DQN baseline, followed by the QoE-oriented trajectory optimization and the bandwidth-oriented trajectory optimization in a stepwise manner. This separation allows us to distinguish the improvement introduced by CRR from the additional gains contributed by trajectory-level data optimization.

3.4 Per-Step Decision Analysis

To gain deeper insight into the sequential decision patterns of each algorithm, we analyze their action–reward trajectories on a representative streaming trace, as illustrated in Fig. 8. Note that results for Mpc+fixed-preload are not included due to

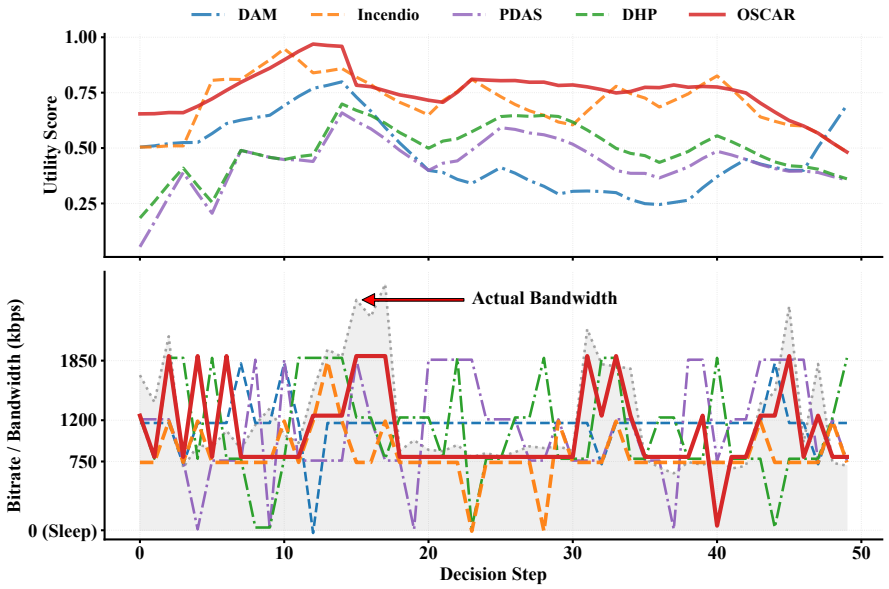

Fig. 8 Action-reward trajectories of different algorithms on a single representative streaming trace.

.

its significantly inferior performance. The figure aggregates bitrate selections across all videos in the playback queue and plots them against the dynamically varying available bandwidth. This unified view clearly reveals each algorithm's bitrate adaptation behavior, avoiding excessive detail from per-video visualization while highlighting distinct decision patterns that directly explain their overall performance differences.

Incendio exhibits a consistently conservative behavior, favoring low bitrates throughout the entire session regardless of available bandwidth. While this ultra-conservative approach minimizes bandwidth waste and rebuffering risk, it comes at the cost of sacrificing visual quality, resulting in the lowest overall QoE among all compared methods. Rule-based methods PDAS and DHP exhibit fundamentally different yet equally problematic behaviors: high volatility in bitrate selection. Both algorithms react aggressively to short-term fluctuations in network conditions, with their decisions closely tracking every dip and surge in bandwidth. Such instability directly impairs streaming smoothness, as frequent bitrate changes introduce perceptible quality variations that degrade user experience despite their intent to maximize instantaneous quality. In stark contrast, DAM demonstrates a rigid commitment to medium bitrates across varying network conditions. This fixed-preference strategy provides a reasonable quality baseline and avoids the volatility seen in rule-based methods. However, its inflexibility proves problematic during low-bandwidth periods: by persisting with medium bitrate selections when network capacity is insufficient, DAM extends chunk download times and increases rebuffering risk—a direct consequence of its inability to adapt downward when conditions deteriorate.

OSCAR overcomes the limitations of prior methods through a context-aware decision policy. Departing from the rigid bitrate preferences and reactive adaptations

Table 2 Inference latency and model complexity of OSCAR and existing SABR algorithms

Method	Mean Latency (ms)	Policy Parameters	FP32 Weights (MiB)
Mpc+fixed-preload	1.412	–	–
DAM	**0.462**	16,784	0.064
PDAS	61.007	–	–
Incendio	4.325	94,877	0.362
DHP	2.391	–	–
OSCAR	1.576	305,296	1.165

seen in existing approaches, OSCAR dynamically tailors its decisions by continuously assessing real-time network conditions, buffer occupancy, and user-watch probability. When bandwidth and buffer conditions permit, it elevates bitrates to sustain high-quality playback; when prefetching becomes unnecessary or wasteful, it promptly pauses or reduces bitrates. This balanced strategy combines the adaptability of learning-based methods with the stability of optimized rule-based control, directly explaining its superior end-to-end utility as demonstrated in the full evaluation.

3.5 Inference Time

To evaluate the computational overhead of OSCAR, we measure its per-decision inference latency and compare it with Mpc+fixed-preload, DAM, PDAS, Incendio, and DHP. All experiments are conducted on an Intel(R) Core(TM) i5-10300H CPU @ 2.50 GHz using a single CPU thread, with GPU acceleration disabled. For neural-network-based methods, inference is performed in FP32 with a batch size of one.

The timing interval covers the computation required for each streaming decision, including state processing, model or heuristic computation, action selection, and action decoding. Initialization, simulator advancement, logging, and other I/O operations are excluded because they are not part of the online decision computation. To avoid initialization effects, the first 10 inference calls on each trace are discarded as warm-up, and the reported latency is averaged over the remaining decisions across the evaluated traces.

As shown in Table 2, OSCAR requires only 1.576 ms on average to generate each streaming decision on CPU. Although DAM achieves the lowest inference latency at 0.462 ms, OSCAR remains substantially faster than Incendio (4.325 ms) and PDAS (61.007 ms), while its latency is comparable to the lightweight heuristic methods Mpc+fixed-preload (1.412 ms) and DHP (2.391 ms). These results indicate that the additional temporal representation and offline policy-learning components of OSCAR introduce modest runtime overhead and do not require expensive online optimization.

We further report the policy size of the learning-based methods to characterize their model complexity. OSCAR contains 305,296 trainable parameters, corresponding to approximately 1.165 MiB of FP32 policy weights. In comparison, DAM and Incendio contain 16,784 and 94,877 parameters, requiring approximately 0.064 MiB and 0.362

Table 3 Ablation study of key components in OSCAR.

Configuration	Average Utility Score	Average QoE	Average Bandwidth Wastage
DQN	24.67	33.25	28.37%
DQN+CRR	25.48	34.24	26.1%
DQN+CRR+QoE-Opt	26.33	36.15	26.5%
OSCAR	**28.41**	**36.27**	**23.2%**

MiB, respectively. Although OSCAR employs a larger policy network, its absolute memory footprint remains small, and its mean CPU inference latency remains below 2 ms per decision.

It is also important to note that trajectory optimization and CRR training are performed entirely offline. They introduce no additional trajectory-repair or CRR-specific computation during deployment. Once training is completed, online inference only requires a forward pass through the learned policy to produce the SABR action. Therefore, the improvements introduced by OSCAR's offline learning pipeline do not translate into additional online optimization stages.

3.6 Deep Dive

We further investigate the design of OSCAR through a series of controlled ablation studies on the DUASVS and Oboe/FCC datasets. Specifically, we first quantify the contribution of each major component in OSCAR and examine the sensitivity to the conservative buffer threshold. We then evaluate the choice of temporal encoder and the advantage-weighting function used by CRR. Unless otherwise stated, all variants use the same training/test split and are evaluated on the same held-out video–network pairs.

3.6.1 Component Ablation

We first perform a stepwise ablation to quantify the contribution of the three major components in OSCAR. Starting from a vanilla DQN baseline trained directly on the raw logged trajectories, we first introduce CRR as the offline policy-improvement mechanism. We then add the QoE-oriented trajectory optimization (QoE-Opt) and finally the bandwidth-oriented trajectory optimization to obtain the complete OSCAR framework.

As shown in Table 3, vanilla DQN obtains an average utility score of 24.67, an average QoE of 33.25, and bandwidth wastage of 28.37%. Direct value-based learning from a fixed behavioral dataset is vulnerable to distributional shift and to suboptimal decisions contained in the logged trajectories.

Adding CRR increases the utility score from 24.67 to 25.48, while increasing QoE from 33.25 to 34.24 and reducing bandwidth wastage from 28.37% to 26.1%. CRR performs advantage-weighted behavioral regression and allows only logged actions with positive estimated advantage to contribute to the policy-improvement objective.

Table 4 Impact of the conservative buffer threshold b_i^{th}.

Conservative Threshold	Average Utility Score	Average QoE	Average Bandwidth Wastage
$b_i^{th} = 0.5\,\mathrm{s}$	26.31	33.84	**22.6%**
$b_i^{th} = 1.0\,\mathrm{s}$	**28.41**	**36.27**	23.2%
$b_i^{th} = 1.5\,\mathrm{s}$	26.73	36.10	25.1%

The improvement therefore demonstrates the benefit of constraining offline policy optimization toward advantageous actions supported by the behavioral data.

Adding QoE-Opt further increases the utility score from 25.48 to 26.33 and substantially improves QoE from 34.24 to 36.15. Meanwhile, bandwidth wastage slightly increases from 26.1% to 26.5%. This trade-off is consistent with the purpose of QoE-Opt: it repairs conservative bitrate decisions when a higher-bitrate alternative remains feasible according to the post-download buffer and temporal slack conditions. The resulting trajectories encourage higher-quality playback, but the more aggressive quality decisions can introduce a small increase in bandwidth consumption.

Finally, incorporating the bandwidth-oriented trajectory optimization produces the complete OSCAR framework. The average QoE remains nearly unchanged at 36.27, while bandwidth wastage decreases substantially from 26.5% to 23.2%. Consequently, the overall utility increases from 26.33 to 28.41. Compared with the vanilla DQN baseline, full OSCAR improves average utility by approximately 15.2%. These results show that the two trajectory-optimization objectives play complementary roles: QoE-Opt primarily improves playback quality, whereas bandwidth-oriented optimization suppresses unnecessary prefetching and recovers bandwidth efficiency.

3.6.2 Sensitivity to the Conservative Buffer Threshold

We next evaluate the sensitivity of the configurable conservative buffer threshold b_i^{th} used by the bandwidth-oriented trajectory optimization. As shown in Table 4, setting $b_i^{th} = 0.5\,\mathrm{s}$ results in the lowest bandwidth wastage of 22.6%, but reduces the average QoE to 33.84. A small threshold causes the curation rule to suppress prefetching more aggressively, which conserves bandwidth but also reduces the amount of buffered high-quality content available for playback.

Increasing the threshold to 1.5 s relaxes the bandwidth control and raises the QoE to 36.10, but bandwidth wastage increases to 25.1%. The default setting of $b_i^{th} = 1.0\,\mathrm{s}$ achieves the highest overall utility of 28.41 by balancing these two objectives, with an average QoE of 36.27 and bandwidth wastage of 23.2%. This result also shows that b_i^{th} controls a genuine quality–bandwidth trade-off rather than monotonically improving either individual metric.

3.6.3 Temporal Encoder Ablation

Although each SABR state already contains a short window of recent throughput observations, a single state cannot explicitly summarize the temporal evolution of

Table 5 Ablation of temporal encoders. All variants use the same trajectory-optimized dataset and CRR objective.

Encoder	Average Utility Score	Average QoE	Bandwidth Wastage	Policy Parameters	Mean Latency (ms)
Single-step MLP	26.71	35.04	25.6%	297,312	1.083
5-state stacked MLP	27.39	35.55	24.6%	303,408	1.296
GRU	27.88	35.91	23.9%	301,728	1.438
LSTM	**28.41**	**36.27**	**23.2%**	305,296	1.576

multi-video buffer occupancy, previous prefetching decisions, and user-watch probabilities across consecutive decision steps. We therefore compare the LSTM encoder used by OSCAR with three simpler alternatives: a single-step SABR-state MLP, an MLP that concatenates the five most recent SABR states, and a GRU.

To avoid attributing performance differences merely to model capacity, we adjust the hidden dimensions of the encoder variants to maintain comparable parameter counts. All variants use the same trajectory-optimized dataset, binary CRR objective, policy/value heads, training budget, and held-out evaluation sessions. We additionally report the mean CPU inference latency to evaluate the performance gain together with its computational cost.

Table 5 shows a clear benefit from modeling information across consecutive SABR decisions. The single-step MLP achieves an average utility score of 26.71. Stacking the five most recent states increases the utility to 27.39 and reduces bandwidth wastage from 25.6% to 24.6%. This improvement indicates that the temporal evolution across full SABR states contains useful information beyond the throughput history already embedded in an individual state.

The GRU further increases utility to 27.88 and reduces bandwidth wastage to 23.9%. Unlike fixed state concatenation, its recurrent hidden state can selectively summarize information accumulated across previous decisions rather than treating a fixed number of historical states independently.

The LSTM achieves the best overall performance, with an average utility score of 28.41, an average QoE of 36.27, and bandwidth wastage of 23.2%. Its utility is approximately 6.4%, 3.7%, and 1.9% higher than those of the single-step MLP, the 5-state stacked MLP, and the GRU, respectively. Since all four encoders have comparable parameter counts, the improvement over the stacked MLP cannot be explained simply by a larger network or by exposing the model to additional historical observations.

Instead, the result suggests that learnable recurrent state aggregation is useful for capturing coupled temporal dependencies among throughput trends, the evolution of multiple playback and prefetch buffers, previous download decisions, and changing user-watch probabilities. The additional computational cost is modest: the LSTM requires 1.576 ms per decision on average, compared with 1.083–1.438 ms for the alternative encoders. We therefore retain the LSTM in the final OSCAR configuration.

Table 6 Impact of different CRR advantage-weighting functions.

Weighting	Average Utility Score	Average QoE	Bandwidth Wastage
Exponential ($\beta = 1.0$)	27.69	**36.44**	25.2%
Sigmoid ($\beta = 0.5$)	28.08	36.22	23.8%
Binary	**28.41**	36.27	**23.2%**

3.6.4 CRR Weighting Ablation

The original CRR formulation permits different functions for weighting behavioral actions according to their estimated advantages. OSCAR adopts binary weighting,

$$w_{\text{bin}}(s, a) = \mathbb{I}[A(s, a) > 0], \tag{19}$$

which allows positive-advantage actions to contribute to the policy-improvement objective and assigns zero weight to non-positive-advantage actions. To examine whether this hard selection is preferable to softer alternatives, we additionally evaluate clipped exponential weighting,

$$w_{\text{exp}}(s, a) = \min\left\{\exp\left(\frac{A(s, a)}{\beta}\right), w_{\max}\right\}, \tag{20}$$

and sigmoid weighting,

$$w_{\text{sig}}(s, a) = \sigma\left(\frac{A(s, a)}{\beta}\right). \tag{21}$$

For the two soft variants, the temperature parameter β is selected using only the validation set, while the dataset, critic, temporal encoder, policy/value heads, and optimization settings are held fixed. The selected temperatures are $\beta = 1.0$ for exponential weighting and $\beta = 0.5$ for sigmoid weighting.

As shown in Table 6, the three weighting strategies exhibit different trade-offs between playback quality and bandwidth efficiency. Exponential weighting achieves the highest average QoE of 36.44, but also increases bandwidth wastage to 25.2%, leading to a lower overall utility score of 27.69.

One possible explanation is that exponential weighting depends strongly on the magnitude of the estimated advantage. A relatively small number of actions with large positive estimated advantages can therefore receive disproportionately high weights. While this can favor more aggressive high-quality decisions, errors in advantage estimation may also amplify unnecessary bitrate upgrades or prefetching actions, which is consistent with the higher bandwidth wastage observed in this variant.

Sigmoid weighting provides a bounded and smoother alternative, achieving an average utility score of 28.08, an average QoE of 36.22, and bandwidth wastage of 23.8%. However, sigmoid weighting assigns nonzero weights even to actions with mildly negative estimated advantages. Such actions can therefore continue to influence the policy update, retaining part of the suboptimal behavior contained in the offline trajectories.

Binary weighting achieves the highest overall utility of 28.41 and the lowest bandwidth wastage of 23.2%, while maintaining a comparable QoE of 36.27. Unlike

exponential weighting, its update depends only on the sign of the estimated advantage rather than its magnitude, avoiding the direct amplification of potentially inaccurate large advantage estimates. Unlike sigmoid weighting, it completely prevents non-positive-advantage actions from contributing to the policy-improvement objective.

These results indicate that binary weighting provides the most favorable QoE–bandwidth trade-off among the evaluated CRR variants on the curated OSCAR dataset. We therefore adopt binary advantage weighting in the final OSCAR configuration.

4 Conclusion

In this paper, we present OSCAR, a trajectory-optimized offline reinforcement learning framework for short-video adaptive bitrate streaming. OSCAR constructs a high-quality offline dataset by aggregating and filtering trajectories from existing SABR algorithms under diverse network conditions and user behaviors, thereby eliminating the need for costly online exploration. To address the inherent distributional shift in offline RL, OSCAR adopts a distributional DQN enhanced with an LSTM for temporal feature learning, combined with critic-regularized regression to ensure stable and robust policy optimization. Extensive experimental results demonstrate that OSCAR outperforms state-of-the-art methods across various network environments and user preference patterns, achieving superior overall utility while effectively balancing streaming quality and bandwidth efficiency.

Statements and Declarations

Funding

This work was supported in part by the National Natural Science Foundation of China (62471215 and 62431011).

Competing Interests

The authors have no competing interests to declare that are relevant to the content of this article.

Data Availability

The MMGC2022 short-video streaming simulator and its associated data are publicly available from the ACM Multimedia 2022 Short Video Streaming Challenge repository: https://github.com/AItransCompetition/Short-Video-Streaming-Challenge.

The Oboe bandwidth traces used in this study are publicly available from the Oboe repository: https://github.com/USC-NSL/Oboe, and are described in Ref. [3].

The FCC bandwidth traces originate from the Measuring Broadband America 2016 dataset provided by the Federal Communications Commission: https://www.fcc.gov/reports-research/reports/measuring-broadband-america/raw-data-measuring-broadband-america-2016.

References

[1] Huang, T., Zhou, C., Zhang, R.-X., Wu, C., Yao, X., Sun, L.: Comyco: Quality-aware adaptive video streaming via imitation learning. In: Proceedings of the 27th ACM International Conference on Multimedia, pp. 429–437 (2019)

[2] Yin, X., Jindal, A., Sekar, V., Sinopoli, B.: A control-theoretic approach for dynamic adaptive video streaming over http. In: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, pp. 325–338 (2015)

[3] Akhtar, Z., Nam, Y.S., Govindan, R., Rao, S., Chen, J., Katz-Bassett, E., Ribeiro, B., Zhan, J., Zhang, H.: Oboe: Auto-tuning video abr algorithms to network conditions. In: Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, pp. 44–58 (2018)

[4] Ran, D., Hong, H., Chen, Y., Ma, B., Zhang, Y., Zhao, P., Bian, K.: Preference-aware dynamic bitrate adaptation for mobile short-form video feed streaming. IEEE Access **8**, 220083–220094 (2020)

[5] Zhang, G., Liu, K., Hu, H., Guo, J.: Short video streaming with data wastage awareness. In: 2021 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6 (2021). IEEE

[6] Zhang, H., Ban, Y., Zhang, X., Guo, Z., Xu, Z., Meng, S., Li, J., Wang, Y.: Apl: Adaptive preloading of short video with lyapunov optimization. In: 2020 IEEE International Conference on Visual Communications and Image Processing (VCIP), pp. 13–16 (2020). IEEE

[7] Mao, H., Netravali, R., Alizadeh, M.: Neural adaptive video streaming with pensieve. In: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 197–210 (2017)

[8] Tang, Y., Agrawal, S.: Discretizing continuous action space for on-policy optimization. In: Proceedings of the Aaai Conference on Artificial Intelligence, vol. 34, pp. 5981–5988 (2020)

[9] Fan, T.-H., Wang, Y.: Soft actor-critic with integer actions. In: 2022 American Control Conference (ACC), pp. 2611–2616 (2022). IEEE

[10] Zhou, C., Ban, Y., Zhao, Y., Guo, L., Yu, B.: Pdas: Probability-driven adaptive streaming for short video. In: Proceedings of the 30th ACM International Conference on Multimedia, pp. 7021–7025 (2022)

[11] Qian, S.-Z., Xie, Y., Pan, Z., Zhang, Y., Lin, T.: Dam: Deep reinforcement learning based preload algorithm with action masking for short video streaming. In: Proceedings of the 30th ACM International Conference on Multimedia, pp.

7030–7034 (2022)

[12] Li, Y., Zheng, Q., Zhang, Z., Chen, H., Ma, Z.: Improving abr performance for short video streaming using multi-agent reinforcement learning with expert guidance. In: Proceedings of the 33rd Workshop on Network and Operating System Support for Digital Audio and Video, pp. 58–64 (2023)

[13] Zhang, H., Zhou, A., Lu, J., Ma, R., Hu, Y., Li, C., Zhang, X., Ma, H., Chen, X.: Onrl: improving mobile video telephony via online reinforcement learning. In: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking. MobiCom '20. Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3372224.3419186 . https://doi.org/10.1145/3372224.3419186

[14] Zheng, Q., Chen, H., Ma, Z.: Bamboo: Boosting training efficiency for real-time video streaming via online grouped federated transfer learning. In: Proceedings of the 7th Asia-Pacific Workshop on Networking. APNet '23, pp. 215–216. Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3600061.3600069 . https://doi.org/10.1145/3600061.3600069

[15] Levine, S., Kumar, A., Tucker, G., Fu, J.: Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643 (2020)

[16] Jia, L., Zhou, C., Huang, T., Li, C., Sun, L.: Dancing with shackles, meet the challenge of industrial adaptive streaming via offline reinforcement learning. In: IEEE INFOCOM 2024-IEEE Conference on Computer Communications, pp. 2169–2178 (2024). IEEE

[17] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., *et al.*: Human-level control through deep reinforcement learning. nature **518**(7540), 529–533 (2015)

[18] Gao, W., Zhang, L., Yang, H., Zhang, Y., Yan, J., Lin, T.: Dhp: A joint video download and dynamic bitrate adaptation algorithm for short video streaming. In: International Conference on Multimedia Modeling, pp. 587–598 (2023). Springer

[19] Burger, E.W., Krishnaswamy, P., Schulzrinne, H.: Measuring broadband america: A retrospective on origins, achievements, and challenges. ACM SIGCOMM Computer Communication Review **53**(2), 11–21 (2023)

[20] Zhang, G., Zhang, J., Liu, K., Guo, J., Lee, J.Y., Hu, H., Aggarwal, V.: Duasvs: A mobile data saving strategy in short-form video streaming. IEEE Transactions on Services Computing **16**(2), 1066–1078 (2022)

[21] Zuo, X., Li, Y., Xu, M., Ooi, W.T., Liu, J., Jiang, J., Zhang, X., Zheng, K., Cui, Y.: Bandwidth-efficient multi-video prefetching for short video streaming.

In: Proceedings of the 30th ACM International Conference on Multimedia, pp. 7084–7088 (2022)