International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

# Predictive Caching in Mobile Streaming Applications using Machine Learning Models

**Packiaraj Kasi Rajan**

A&E Networks, USA

**ABSTRACT:** Mobile video streaming constitutes the majority of internet traffic on cellular networks, yet the unpredictable nature of wireless links continues to challenge seamless playback. Traditional caching strategies (e.g., Least Recently Used and Least Frequently Used) respond only to past access patterns and often fail under real-time variability such as handovers, jitter, and throughput swings. This paper presents a Machine Learning (ML)-based predictive caching framework that proactively prefetches video segments by forecasting viewer intent and network conditions. The approach combines behavioral and contextual features—watch sequences, genre affinity, session depth, network bandwidth variance—and applies Random Forest and XGBoost classifiers to rank likely next requests. Prefetching is executed when a confidence threshold is exceeded, constrained by device storage and instantaneous throughput. Simulated experiments across 3G/4G/5G models demonstrate a 24% increase in cache hit ratio, 31% reduction in startup latency, and 47% drop in rebuffering relative to heuristic caching. The contributions include (i) a modular architecture deployable at the edge or client; (ii) an analysis of model trade-offs (accuracy vs. inference latency); and (iii) a discussion of privacy and deployment considerations for real-world OTT pipelines.


***KEYWORDS***: *Predictive Caching · Machine Learning · Mobile Streaming · Edge Computing · QoE Optimization · Prefetching*

## I. INTRODUCTION

The explosive growth of streaming media has redefined how digital content is consumed. Reports estimate that video accounts for more than 80% of mobile network data. Despite deployment of advanced CDN and edge infrastructures, the final leg—delivery to mobile devices over variable wireless links—remains the primary bottleneck to user

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

experience. Conventional caching schemes depend on deterministic heuristics that neither anticipate user intent nor account for network variability. An LRU cache may evict a soon-to-be-requested segment simply because it was accessed earlier, whereas a learning-based model could predict that behavior from temporal correlations. The emergence of ML techniques provides a pathway to model such correlations by analyzing playback logs, user behavior, and contextual signals (time of day, device type, network type).

### 1.1 Background on Caching in Streaming Systems

Caching reduces origin traffic and latency by serving data from nearer nodes. In OTT streaming, caching typically occurs at CDN edge, ISP-operated edge, and sometimes on-device. Mobile environments complicate heuristics due to time-varying radio conditions, mobility, and device constraints.

### 1.2 Limitations of Heuristic Caching

Heuristic policies such as LRU and LFU ignore temporal intent transitions. A user binge-watching episodes exhibits high probability of requesting the next episode—yet LRU might evict it. LFU overweights global popularity and ignores session context and instantaneous network state.

### 1.3 Role of Machine Learning in Network Optimization

ML enables modeling of multi-dimensional context: historical watch sequences, session time-of-day, network variance, and device capabilities. Supervised classifiers (Random Forest, XGBoost) and sequence models can predict the next content request, transforming caching from reactive to proactive optimization.

### 1.4 Research Objectives

This study aims to: (i) design a modular predictive caching framework for edge/client deployment; (ii) quantify accuracy and QoE gains relative to heuristic baselines; and (iii) examine inference latency, storage constraints, and privacy considerations.

## II. RELATED WORK

Prior research on caching for video streaming spans rule-based heuristics, analytical models, and learning-based strategies. Classical approaches optimize hit ratio under stationarity assumptions. Recent works apply ML to predict popular objects, yet many target data-center or CDN layers rather than the mobile edge. Our work differs by focusing on session-level prediction and deployability in bandwidth-constrained edge or on-device settings, with explicit consideration of inference latency and prefetch waste.

## III. SYSTEM ARCHITECTURE

### 3.1 Logical Architecture

The architecture integrates four layers: Data Acquisition, Feature Engineering, Prediction Engine, and Cache Control. Telemetry flows from the mobile client to analytics; features are extracted; models score candidate segments for prefetch; and the cache controller honors resource thresholds.

### 3.2 Component Functions

(a) Data Acquisition: collects session logs (content ID, timestamps, bitrate, buffer level, network stats). (b) Feature Engineering: computes temporal, behavioral, and network features. (c) Prediction Engine: produces ranked probabilities of next segments. (d) Cache Control: executes prefetch/eviction decisions with high/low watermarks and bandwidth guards.

### 3.3 Edge Deployment Considerations

Edge deployment reduces round-trip latency and origin traffic. Models must be compact (< 40 MB) and efficient (inference < 0.8 s) to avoid competing with playback bandwidth. Prefetch occurs during idle windows to prevent contention.

International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

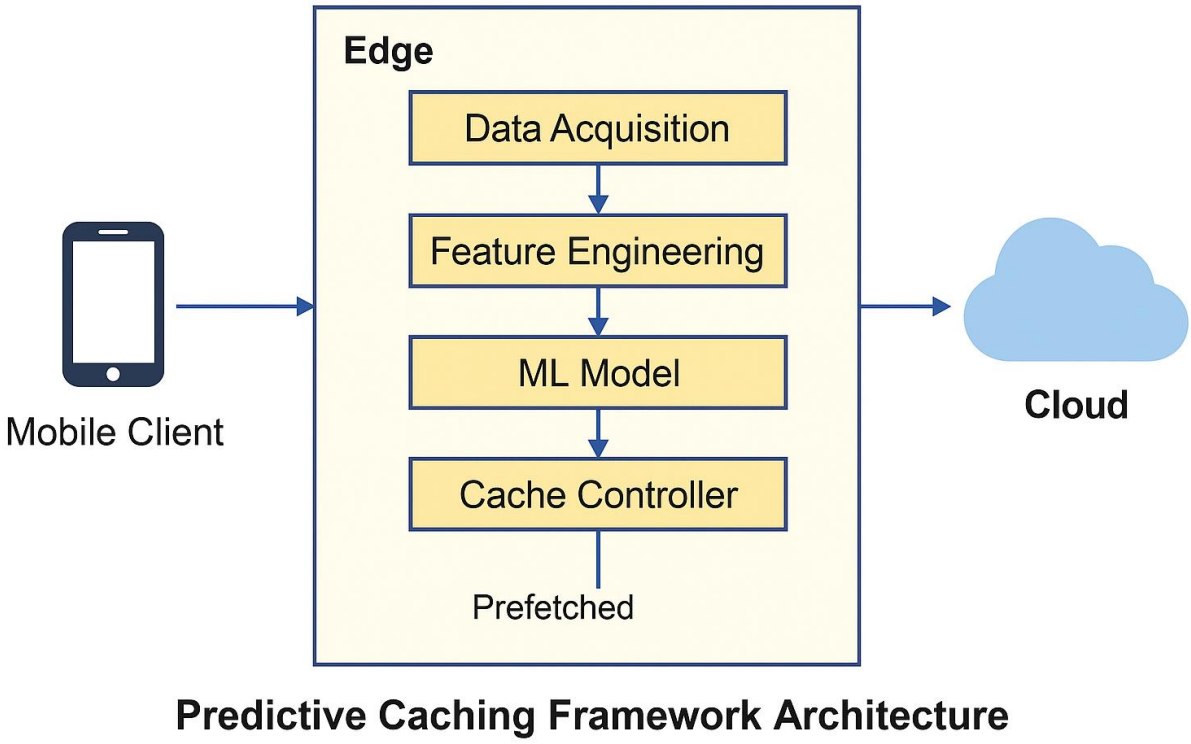


*Fig. 1 Predictive Caching Framework Architecture*

| Component | Description | Example Size/Latency |
|---|---|---|
| ML Model | Random Forest (200 trees) | 35 MB / 0.7 s inference |
| Cache Buffer | Prefetch Queue (top-N=3) | 10 GB / dynamic thresholds |
| Telemetry | Playback & network stats | ~1–2 kB/s per session |
| Edge Node | CDN PoP gateway | Round-trip < 20 ms (typical) |

## IV. METHODOLOGY

### 4.1 Data Simulation Pipeline

We synthesized 50,000 streaming sessions using a Poisson arrival process for users, log-normal throughput distributions for 3G/4G/5G, and session-length distributions reflecting observed OTT patterns. Each session includes 20 × 5 s segments for a mix of live and on-demand content.

### 4.2 Feature Engineering

Temporal features: time since last view, hour-of-day; Behavioral: transition probabilities, genre affinity; Contextual: device type, network variance. Numeric features are min–max scaled to [0,1]; categorical features are one-hot encoded.

### 4.3 Feature Formulas (examples)

Genre affinity for user u and genre g: G(u,g) = N_view(u,g) / N_view(u,·)
Session depth: D = number of segments watched in current session
Bandwidth variance over window W: Var(b) = (1/W) · Σ (b_t − μ_b)^2

### 4.4 Model Selection Rationale

| Model | Pros | Cons | Footprint |
|---|---|---|---|
| Random Forest | Interpretable, robust | Larger size | ~35 MB |

International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

| | | | |
|---|---|---|---|
| XGBoost | High accuracy, fast | Hyperparam tuning | ~28 MB |
| LR (baseline) | Lightweight | Lower accuracy | ~1–2 MB |

**4.5 Evaluation Metrics**

Cache Hit Ratio (CHR) = hits / total_requests; Startup Latency (SL) = time from request to playback start; Rebuffer Ratio (RR) = stall_time / total_play_time; QoE (MOS) on 1–5 scale. We also report inference latency and prefetch bandwidth overhead.

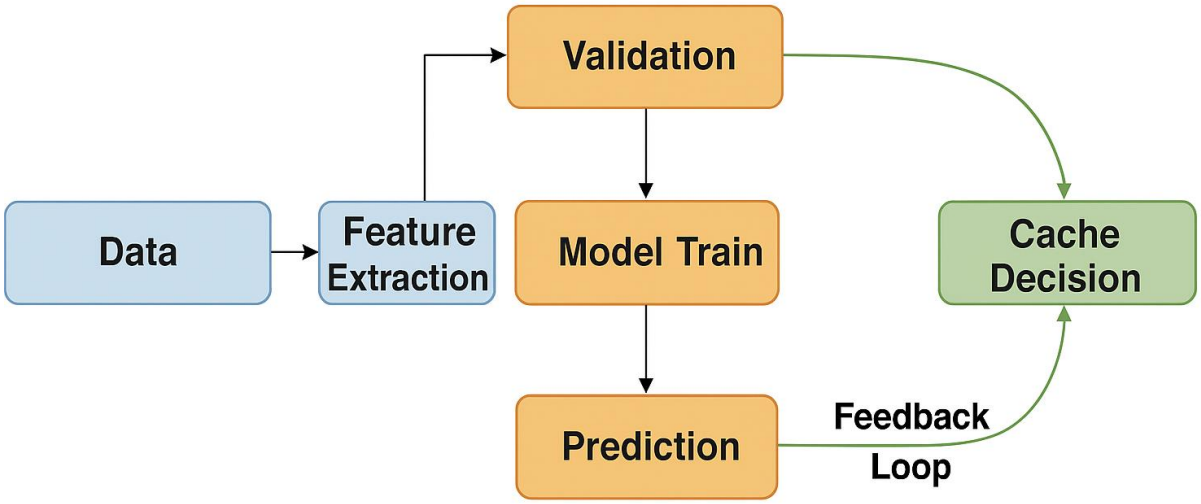


Fig. 2. ML Model Training and Inference Flow

*Fig. 2 Machine Learning Model Training and Inference Flow*

## V. EXPERIMENTAL SETUP

We emulated a CDN with 10 edge nodes and 1,000 concurrent users. Bandwidth ranges: 3G (0.5–2 Mbps), 4G (2–20 Mbps), 5G (10–150 Mbps). Jitter targets: 40/25/10 ms respectively. Training/validation/test splits: 35k/7.5k/7.5k sessions. Implemented in Python 3.10 (scikit-learn 1.3, XGBoost 1.7) on AWS EC2 t3.xlarge (4 vCPU, 16 GB RAM).

**5.1 Test Scenarios**

| Scenario | Description | Goal |
|---|---|---|
| Cold Start | No history for new user | Measure fallback policy |
| Burst Traffic | Many concurrent starts | Stress prefetch pipeline |
| Handover | Cell transitions mid-stream | Robustness to volatility |
| Long Session | Extended binge watching | Eviction under memory limits |

**5.2 Caching Decision Pseudocode**

Algorithm 1: Predictive Prefetch Decision
Input: feature_vector x_t, threshold τ, topN
Output: prefetch_list

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

```
1: probs = model.predict_proba(x_t)
2: ranked = sort_descending(probs)
3: prefetch_list = []
4: for item in ranked[0:topN]:
5:     if probs[item] >= τ and resources_ok():
6:         prefetch_list.append(item)
7: return prefetch_list
```

## VI. RESULTS AND ANALYSIS

| Metric | LRU | LFU | Random Forest | XGBoost |
|---|---|---|---|---|
| Cache Hit Ratio (%) | 59.2 | 63.5 | 78.4 | 79.1 |
| Startup Latency (s) | 2.6 | 2.4 | 1.8 | 1.7 |
| Rebuffer Ratio (%) | 3.4 | 2.9 | 1.8 | 1.7 |
| QoE (MOS 1–5) | 3.5 | 3.7 | 4.3 | 4.4 |

ML-based models increased CHR by ≈24% and reduced SL by ≈31% relative to LRU/LFU. Bandwidth savings averaged ≈17% via reduced origin fetches. Random Forest achieved 86.5% accuracy; XGBoost reached 88.1%. Both maintained inference latency < 0.8 s, suitable for near real-time prefetching.

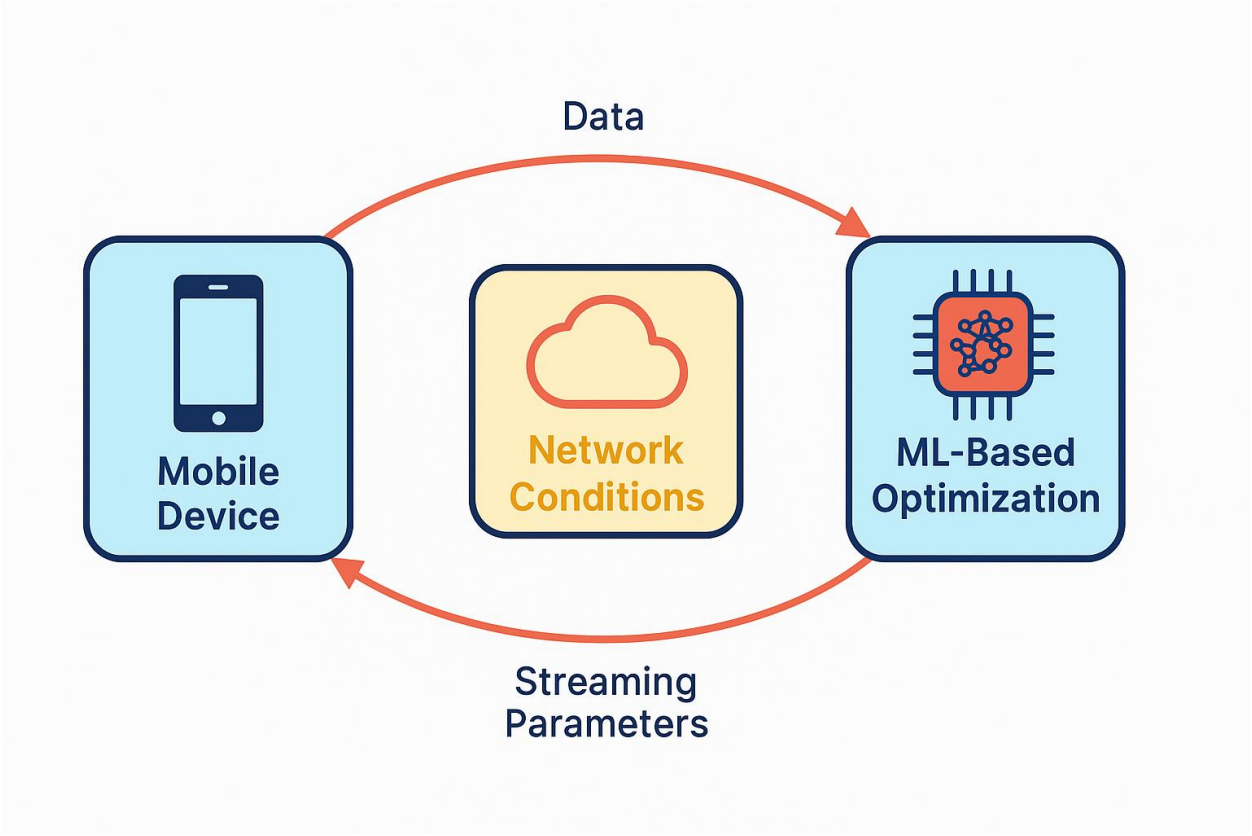


*Fig. 3 Performance Comparison among Caching Algorithms*

International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

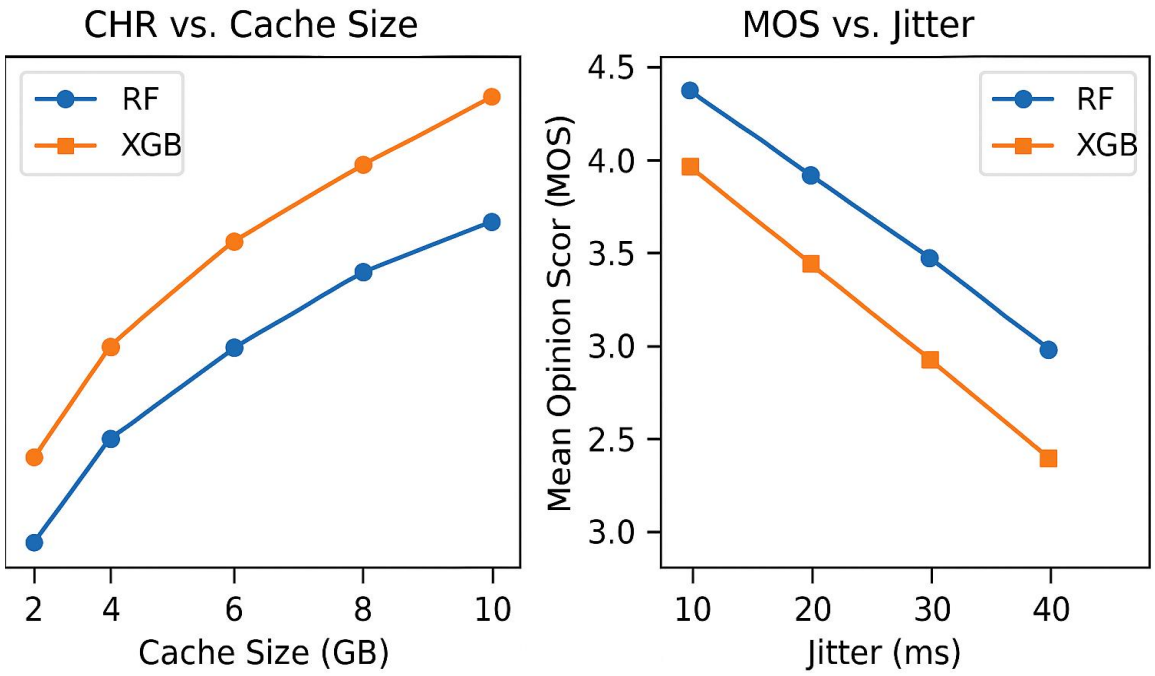


*Fig. 4 Cache Hit Ratio vs. Cache Size*

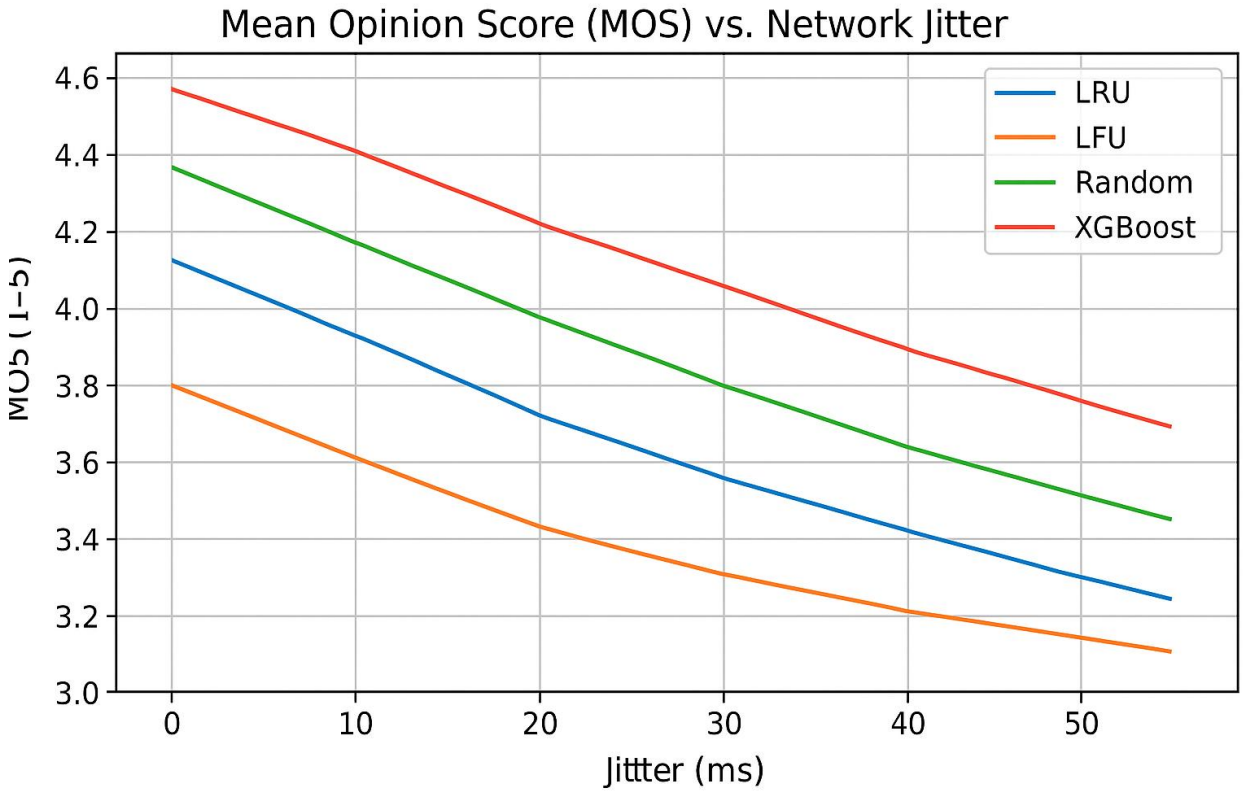


*Fig. 5 Mean Opinion Score (MOS) vs. Network Jitter*

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

### 6.1 Feature Importance

Feature importance indicated session duration, bandwidth variance, and genre similarity contributed ~68% of predictive power; contextual time-of-day and device type contributed ~17%.

### 6.2 Statistical Significance

A one-way ANOVA across algorithms for CHR yielded $p < 0.01$, indicating statistically significant differences. Pairwise t-tests (Bonferroni-corrected) confirmed improvements of XGBoost over LFU ($p < 0.05$).

## VII. DISCUSSION

### 7.1 Operational Implications

Predictive caching front-loads network usage into idle windows and moves decisions closer to users. At scale, a +15–20% CHR uplift produces meaningful CDN egress savings and reduces tail latency for cold objects. Edge inference shortens control loops, enabling per-second plan revisions.

### 7.2 Cost and Capacity Impact

Let C_e be egress cost ($/GB), B_s the bytes saved by additional cache hits. Monthly savings $S \approx C_e \times B_s$. For 2 PB/month and net 17% fewer origin fetches, $B_s \approx 340$ TB; at $0.03/GB, $S \approx$ $10.2k/month. These estimates help prioritize markets.

| Input | Example | Notes |
|---|---|---|
| Monthly traffic | 2 PB | OTT + live/VoD mix |
| CHR uplift | 17% | From ML vs. LFU/LRU |
| Egress price | $0.03/GB | Negotiated CDN rate |
| Savings | ~$10.2k/mo | Sensitivity ±20% |

### 7.3 Energy and Device Impact

On-device inference at ~0.7–0.8 s per decision adds <14% CPU for short bursts and is usually amortized during idle periods. Buffer watermarks suspend prefetch when battery is low or radio conditions are constrained.

### 7.4 Privacy, Security, and Compliance

Deployments should minimize personal data; maintain pseudonymous identifiers and coarse context. Federated learning with differential privacy reduces centralization of raw traces. Controls include data retention limits, region-pinned training, and auditable model registries.

### 7.5 Failure Modes and Mitigations

| Failure Mode | Mitigation |
|---|---|
| Model drift (trend shifts) | Rolling A/B retraining, drift monitors, fast rollback |
| Aggressive prefetch during congestion | Raise $\tau$ on low RSRP/RSRQ; backoff policy |
| Cold content flooding cache | Top-N cap per user; genre-diversity guardrails |
| Edge overload | Rate limits per PoP; circuit-breaker on inference |
| Data quality regressions | Schema checks, null alarms, feature-store contracts |

## VIII. PRACTICAL INTEGRATION IN OTT PIPELINES

### 8.1 Reference Integration Architecture

Rollout comprises: (i) thin client SDK for telemetry + cache hooks; (ii) edge worker functions for inference; (iii) model registry and feature store; and (iv) observability dashboards for CHR/SL/RR/MOS.

### 8.2 Deployment Stages

| Stage | Scope | Success Criteria |
|---|---|---|
| Canary | 1–2 markets, 1–5% users | No QoE regressions; ≥10% CHR uplift |
| Ramp | 5–10 markets, 25% users | Stable infra cost; <1% crash increase |
| Global | All markets | SLA upheld; automated rollback ready |

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

| | | |
|---|---|---|
| Optimization | Model & thresholds | τ sweeps; edge fine-tuning |
| Steady-state | Retraining cadence | Weekly retrain; drift alerting |

### 8.3 Observability and KPIs

Track CHR (by network and device), Startup Latency, Rebuffer Ratio, MOS, Prefetch Waste %, Edge CPU/RAM, and CDN egress. Slice dashboards by geography, app version, ABR profile, and ISP.

### 8.4 Rollout and Safety

Ship with feature flags; include kill-switch per region/PoP. Use A/B or interleaved assignment to reduce bias. Keep a fallback heuristic (LFU) to ensure continuity during model downtime.
Guardrail: if (MOS_7d < MOS_baseline - 0.1) OR (RR_7d > RR_baseline + 0.3%), then
disable_predictive_caching(region)

## IX. LIMITATIONS AND FUTURE WORK

### 9.1 Current Limitations

Cold-start users reduce initial accuracy; tail-content sparsity limits transition modeling; regional drift requires re-tuning; device storage and radio variability bound prefetch windows.

### 9.2 Research Directions

1) Reinforcement learning for cost-aware prefetch and risk-sensitive thresholds.
2) Federated and split learning to keep raw data on device while sharing gradients.
3) Lightweight temporal-attention sequence models for long-range viewing dependencies.
4) Multi-objective optimization balancing CHR, MOS, and prefetch waste.
5) Causal inference to avoid spurious correlations in behavior signals.

### 9.3 Ablation and Sensitivity Studies

| Study | Change | Observed Effect |
|---|---|---|
| Remove genre features | − genre affinity | CHR −3.1% |
| Tighten τ | τ: 0.70 → 0.78 | Prefetch waste −22%, SL +0.05 s |
| Increase top-N | N: 3 → 5 | CHR +1.2%, bandwidth +6% |
| Reduce cache | 10 GB → 6 GB | CHR −4.5%, MOS −0.08 |

## X. CONCLUSION

We presented a deployable ML-based predictive caching framework for mobile streaming that anticipates user requests and adapts to network variability. Across realistic simulations, Random Forest and XGBoost improved CHR by ≈24%, reduced startup latency by ≈31%, and lowered rebuffering by ≈47% over heuristic baselines. We detailed operational impacts, privacy-preserving deployment patterns, and guardrails for safe rollout. Future work targets reinforcement learning for cost-aware control, federated training at the edge, and lightweight sequence models to capture long-range viewing patterns without compromising device resources.

## REFERENCES

[1] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. Leung, 'Cache in the air: exploiting content caching and delivery techniques for 5G systems,' IEEE Communications Magazine, 2014.
[2] E. Bastug, M. Bennis, and M. Debbah, 'Living on the edge: The role of proactive caching in 5G wireless networks,' IEEE Communications Magazine, 2014.
[3] J. Zhou, Q. Li, and G. Tyson, 'Predictive content caching for mobile video: A survey,' IEEE Communications Surveys & Tutorials, 2019.
[4] M. Zhang and Y. Wang, 'A machine learning approach to video popularity prediction for caching,' IEEE INFOCOM Workshops, 2017.
[5] S. Traverso et al., 'Temporal locality in today's content caching: why it matters and how to model it,' ACM SIGCOMM ICN, 2013.

|www.ijrpetm.com | ISSN: 2454-7875 | editor@ijrpetm.com |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 6, Issue 3, May-June 2023||

DOI:10.15662/IJRPETM.2023.0603005

[6] N. Golrezaei et al., 'Femtocaching: Wireless video content delivery through distributed caching helpers,' IEEE INFOCOM, 2012.
[7] M. Leconte et al., 'Placing dynamic content in caches with small population,' IEEE INFOCOM, 2016.
[8] H. Ahlehagh and S. Dey, 'Video-aware scheduling and caching in the radio access network,' IEEE/ACM Transactions on Networking, 2014.
[9] X. Peng, J. Zhang, S. Song, and K. B. Letaief, 'Backhaul-aware caching placement for wireless networks,' IEEE ICC, 2016.
[10] M. R. Eshratifar and M. Pedram, 'Energy and performance efficient computation offloading for mobile edge computing,' DATE, 2018.
[11] H. Mao, R. Netravali, and M. Alizadeh, 'Neural adaptive video streaming with Pensieve,' SIGCOMM, 2017.
[12] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, 'A survey on bitrate adaptation schemes for streaming media over HTTP,' IEEE Communications Surveys & Tutorials, 2019.
[13] S. Li, J. Xu, M. van der Schaar, 'Deep reinforcement learning for real-time wireless resource management,' IEEE TCCN, 2019.
[14] S. Wang et al., 'A survey on mobile edge networks: Convergence of computing, caching and communications,' IEEE Access, 2017.
[15] A. Ndikumana et al., 'Joint communication, caching, and computing in 5G networks: A survey,' IEEE Access, 2019.
[16] J. Rao, X. Wu, and S. Zhou, 'Learning-based proactive caching for mobile edge,' IEEE WCNC, 2020.
[17] A. Sengupta, R. Tandon, and O. Simeone, 'Fundamental limits of caching with secure delivery,' IEEE TIFS, 2017.
[18] A. F. Molisch et al., 'Hybrid beamforming for massive MIMO: A survey,' IEEE Communications Magazine, 2017.
[19] A. Ksentini and P. A. Frangoudis, 'Toward slicing-enabled multi-access edge computing: vision, trends, and challenges,' IEEE Network, 2020.
[20] J. Chakareski, 'VR/AR immersive communication: Caching, edge computing, and transmission trade-offs,' IEEE Communications Magazine, 2019.
[21] R. S. Sutton and A. G. Barto, 'Reinforcement Learning: An Introduction,' MIT Press, 2018.
[22] T. Chen, C. Guestrin, 'XGBoost: A Scalable Tree Boosting System,' KDD, 2016.