Journal of Theoretical and Applied Information Technology
28th February 2026. Vol.104. No.4

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

EDGE-CENTRIC PRIVACY-PRESERVING VIDEO ANALYTICS FOR LATENCY-SENSITIVE MONITORING

DR GAURAV VISHNU LONDHE[1], DR. S. MAHESWARI[2], DR.V.RADHIKA[3], LAVANYA KUMARI PITHANI[4], V SIVARAMARAJU VETUKURI[5], ELANGOVAN MUNIYANDY[6]

[1]Associate Professor, Symbiosis Institute of Technology, Symbiosis International University, Pune, Nagpur Campus, India.
[2]Professor, Department of ECE, Panimalar Engineering College, Chennai, India.
[3]Assistant Professor, Department of CSE-(CyS, DS) and AI&DS, VNR Vignana Jyothi Institute of Engineering and Technology, Hyderabad, India.
[4] Department of Computer Applications, Aditya University, Surampalem, India.
[5]Associate Professor, Department of Computer Science and Engineering, Koneru Lakhmaiah Education Foundation, Vaddeswaram, India.
[6]Department of Biosciences, Saveetha School of Engineering. Saveetha Institute of Medical and Technical Sciences,Chennai,India.

[1]gauravlondhe@gmail.com, [2]maheswarisp@yahoo.co.in, [3]vradhikanaren@gmail.com, [4]lavanyakumari.medisetti@gmail.com,[5]sivaramaraju.vetukuri@gmail.com, [6]muniyandy.e@gmail.com

ABSTRACT

Real-time video analytics at the network edge is essential for smart surveillance, autonomous monitoring, and safety-critical decision making. However, maintaining high detection accuracy under severe latency limits and demonstrable privacy protection is difficult. Raw video transmission on the cloud increases delay and privacy risks, while entirely on-device solutions are limited by computational and energy resources.This paper presents a hybrid edge-centric video analytics architecture that balances latency, accuracy, and privacy via split inference, lightweight on-device processing, and privacy-preserving feature preservation. A compact backbone's early layers do motion filtering, region-of-interest extraction, and feature computing on the device, while a nearby edge server performs deeper inference. Quantization and differential privacy safeguard intermediate characteristics, with selective homomorphic encryption applied to tiny crucial layers for better guarantees. Based on bandwidth and device demand, an adaptive runtime controller chooses the split point and compression level. On VIRAT, PEViD, and Sultani-derived datasets, the suggested technique achieves sub-100 ms median end-to-end latency and near-cloud detection accuracy. Results indicate that modest differential privacy ($\varepsilon \approx 1$) significantly reduces membership-inference attack success with minimal accuracy loss, whereas full homomorphic encryption is too latency-intensive. Split inference and lightweight differential privacy create a realistic and deployable privacy-aware real-time edge video analytics solution.
Keywords: *Video Analytics, Split Inference, Differential Privacy, Homomorphic Encryption, Latency Constraints, Adaptive Controller*

1. INTRODUCTION

As edge cameras are deployed in surveillance and monitoring systems, real-time video analytics with minimal latency and strong privacy protection are needed. Cloud-based techniques have substantial transmission delay and privacy issues, while on-device processing is limited by computational capabilities. This study uses split-inference to balance device and edge server processing to address these issues. The goal is sub-100 ms latency with near-cloud accuracy, and mild differential privacy on intermediate characteristics can limit privacy leakage without affecting real-time performance.[1], [2]. Edge-centric processing shifts early computation to the camera or nearby nodes, reducing round-trip time and network load while offering opportunities to limit exposure of identity-bearing pixels. Recent surveys of edge-based video analytics highlight the practical benefits and engineering challenges of this shift, particularly for latency-sensitive monitoring applications. [3], [4], [5].

Despite progress, a concrete, deployable solution that balances three competing goals, such as strict latency bounds, high detection utility, and quantifiable privacy guarantees, remains elusive. Latency requirements for many monitoring tasks often sit below 100 ms per frame (for example, rapid

alerting in perimeter security or drone-based collision avoidance), while on-device resources are constrained by battery, memory, and limited NPUs [6], [7]. At the same time, transmitting raw frames or high-fidelity features risks exposing personally identifiable information; practical deployments require privacy mechanisms with measurable budgets (e.g., differential privacy ε) or cryptographic assurances [8], [9]when appropriate. The research problem is therefore delivering accurate video analytics under sub-100 ms per-frame latency, under realistic energy and bandwidth caps, while enforcing provable or empirically validated privacy protection.

This work argues for a hybrid approach that places a compact preprocessor and lightweight backbone on-device, transmits protected intermediate features, and completes heavier inference on an edge head. Split inference—partitioning a neural model so early layers run locally, and later layers run remotely—reduces transferred bytes and latency while preserving utility. [10], [11]. Privacy can be addressed at multiple levels: representation abstraction (skeletons, optical flow), lightweight differential-privacy noise injected into intermediate features, and selective use of cryptographic private inference for critical small layers. Prior literature on private inference, split learning, and privacy-aware representations guides these choices but typically sacrifices either latency or formal privacy guarantees; resolving that trade-off in a single, operational pipeline is the principal motivation. [12], [13].

The objective of this research is to design and evaluate an edge-centric video analytics system that attains sub-100 ms per-frame latency, maintains detection accuracy comparable to cloud baselines, and enforces a quantifiable privacy budget.

The paper makes four main contributions:
• A system architecture combining split inference, on-device pre-processing, and a privacy layer that supports DP noise and optional selective cryptographic protection.
• A formal latency–privacy optimization that selects model split point and feature compression under device and network constraints.
• An adaptive runtime controller (pseudocode and implementation) that shifts the split point and compression in response to bandwidth and load.
• An empirical evaluation on standard surveillance and privacy datasets (VIRAT, PEViD, Sultani-derived) reporting latency, mAP/AUC, bandwidth, energy per frame, and privacy-leakage metrics.

The remainder of the paper details related work, the system design and mathematical modelling, experimental methods, and a figure-rich evaluation that quantifies the latency–accuracy–privacy frontier and provides practical rules of thumb for deployment.

The article covers edge video analytics with low latency, accuracy, and privacy. It presents a hybrid split-inference paradigm that processes early neural network layers on-device and completes inference at a nearby edge server utilizing protected intermediate features. Differential privacy is applied to these features to provide measurable privacy protection with minimal impact on accuracy and latency, while selective cryptographic methods illustrate the privacy–latency trade-off. Sub-100 ms latency, near-cloud accuracy, and reduced privacy leakage are achieved with an adaptive controller that dynamically optimizes split point and compression.

2. RELATED WORKS

Edge video analytics has matured rapidly from proof-of-concept pipelines to practical systems that push compute toward cameras and nearby nodes to reduce latency and bandwidth. Recent surveys synthesize this trend, highlighting split computing and on-device pre-processing as effective strategies to lower end-to-end delay and network load while keeping model capacity high at the edge. [14], [15], [16]. At the same time, privacy concerns have spurred a parallel literature that treats raw video as a sensitive stream: publications evaluate representation-level obfuscation (blurring, skeletons, optical flow), algorithmic privacy (differential privacy applied to features or gradients), and cryptographic private inference (HE/MPC) as alternative means to limit information exposure. [17].

Key technical concepts recur across these threads. Split inference (and related split learning) partitions neural networks so that early layers run locally and later layers run remotely; This reduces transmitted bytes and can improve latency when the cut point is chosen adaptively. Differential privacy offers a lightweight, composable mechanism to bound the information leaked by intermediate features or aggregated updates, trading a tunable privacy budget (ε) against utility. Homomorphic encryption and secure multi-party computation provide strong cryptographic guarantees but often impose heavy computational and communication overheads, especially for non-linear layers; hybrid strategies that apply cryptography selectively to small linear

components while using efficient secure protocols for non-linearities aim for practicality [18], [19].

Lightweight vision backbones (MobileNetV3, MobileViT, TinyViT, and windowed Swin variants) and representation abstraction (skeletons, flow, frequency transforms) are repeatedly recommended for resource-limited devices: they preserve task-relevant signals while reducing identity-bearing content. Practical private-inference work emphasizes engineering tradeoffs—latency, energy, and implementation complexity—over purely theoretical guarantees, and several systems papers demonstrate that careful model partitioning and quantisation can yield near-cloud accuracy with much lower latency. [20], [21].

Gaps remain. Heavy cryptographic approaches usually meet privacy targets but fail strict real-time constraints. Simple obfuscation methods can protect identity but degrade utility unpredictably. Few studies jointly optimize latency, accuracy, and provable privacy within an end-to-end edge deployment, and longitudinal privacy accounting for continuous streams receives limited attention. This literature landscape motivates hybrid, adaptive designs that combine representation abstraction, DP noise on intermediate features, and selective cryptographic protection tuned to runtime conditions.

3. METHODOLOGY

In order to provide a framework for real-time edge video analytics that protects users' privacy, this study uses an experimental system-design methodology. The suggested approach takes advantage of split inference, which means that the edge device processes the first neural network layers, while a neighboring edge server processes the deeper layers utilizing intermediate feature representations. To ensure privacy, features are compressed and differentially encrypted; selective homomorphic encryption is reserved for latency-tolerant situations. In response to changes in bandwidth and device load, an adaptive runtime controller chooses the model's split point and compression level on the fly. Latency, accuracy, energy consumption, communication cost, and privacy-attack parameters are used to assess the method using public surveillance datasets

.3.1. Research Design

The study follows a mixed design that combines experimental system engineering with analytical modelling. The experimental component involves implementing a prototype that divides computation between an on-device backbone and an edge server. The analytical component develops equations to represent latency, privacy budgets, and utility trade-offs.

Independent variables include split point, feature compression ratio, privacy mechanism, and available bandwidth. Dependent variables include end-to-end latency, accuracy, transmitted bytes, and privacy leakage under defined attack models. Each configuration is tested on multiple datasets to measure generalisation across scenes and lighting conditions. Performance is summarised using median latency and 95% confidence intervals.

3.2. System Architecture

The proposed system consists of five major modules: on-camera preprocessor, on-device backbone, feature protection module, edge head with decision engine, and a runtime adaptation controller.

3.2.1 On-Camera Preprocessor

This module reduces the volume of input while removing redundant frames. Motion-based filtering and region-of-interest (ROI) extraction limit the number of frames sent to the next stage, which reduces bandwidth consumption and enhances privacy.

3.2.2 On-Device Backbone

A lightweight deep model, such as MobileNetV3 or MobileViT-Tiny, runs the initial layers and produces intermediate feature tensors with low latency. The backbone is selected for its efficiency on NPUs or low-power CPUs commonly found on edge devices.

3.2.3 Feature Protection Module

Intermediate features pass through a protection layer before transmission. Three protection mechanisms are supported:

- Differential privacy noise calibrated for a specified epsilon.
- Quantisation and compression that reduce bandwidth and remove identity traces.
- Selective homomorphic encryption for sensitive linear operations when latency budgets permit.

3.2.4 Edge Head and Decision Engine

Heavier layers execute on the edge server. The decision engine smooths outputs over time and triggers alerts for anomalies or events. Thresholds

and aggregation rules depend on the monitoring application.

3.2.5 Runtime Adaptation Controller

A controller monitors bandwidth, device load, and frame latency. When constraints change, the controller adjusts the split point, compression level, or privacy mode. Figure 1 shows the data flow through the proposed edge-centric privacy-preserving video analytics system.

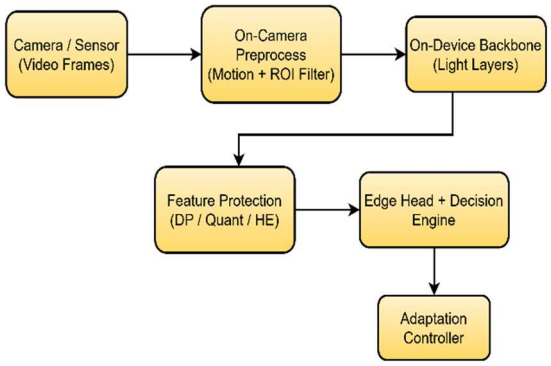

Figure 1. System Architecture

3.3. Dataset Collection and Processing

The dataset collection and processing pipeline uses multiple surveillance datasets such as VIRAT, PEViD, and public anomaly detection collections to ensure robust evaluation. Frames are sampled at application-relevant rates, typically between 5 and 15 fps, and a lightweight detector or motion mask extracts regions of interest to reduce input volume sent to the backbone. To measure privacy–utility trade-offs, parallel representations are produced for each ROI: raw cropped frames, skeleton key points, optical flow maps, and frequency-domain encodings. Standard normalization and low-impact augmentations, such as brightness adjustment, small rotations, and scale jitter, improve model robustness without altering privacy properties. For differential privacy experiments, activation sensitivity is estimated at the chosen split layer to calibrate Gaussian noise correctly, ensuring that the noise scale matches the targeted privacy budget. Faces are blurred when required by the dataset policy. Only synthetic privacy-attack experiments are permitted for datasets lacking explicit consent documentation.

3.4. Sample Selection

Scene-level splitting ensures samples from the same physical location do not appear in both training and testing sets. Stratified sampling across lighting conditions, crowd density, and movement complexity helps avoid bias. Scenarios include crowded streets, indoor hallways, and sparse outdoor environments. Each scenario is evaluated independently and then aggregated to produce summary results.

3.5. Data Analysis Techniques

3.5.1 Model Training

Training occurs in two stages. First, the full network trains on unprotected data to establish the baseline. Second, the model fine-tunes on protected or compressed features to simulate real operating conditions. Loss functions include cross-entropy for classification tasks and joint regression-classification losses for detection tasks. The experimental parameters are shown in Table 1.

Table 1. Experimental Parameters

Parameter	Values Used	Description
Max latency L_{max}	100 ms	Operational latency target
Bandwidth levels	0.5, 5, 50 Mbps	Simulated network tiers
Split points	2, 4, 6	Layer index at which the model is partitioned
Compression ratios	1, 4, 8, 16	Levels of feature compression
Privacy budgets ε	0.5, 1, 5, 10	Differential privacy settings
Frame rates	5, 10, 15 fps	Evaluated scenarios
Batch size	8	Training configuration

3.5.2 Privacy Mechanism Implementation

1) Differential privacy (DP)

Differential privacy protects intermediate activations by adding calibrated random noise so that an attacker cannot reliably tell whether a particular frame influenced the released features. Implementation starts by bounding activation sensitivity, typically by clipping each activation vector or enforcing an L2 norm limit S. Given S and a target privacy budget ε (and a small δ for Gaussian mechanisms), compute the noise scale σ using

standard Gaussian DP formulas or a privacy accountant such as the moment's accountant or Rényi DP to handle composition over many releases. Inject the noise elementwise into the activation tensor or into a compact sketch of the tensor. During training, apply the same noise model so the network learns to tolerate perturbations, and the measured accuracy reflects runtime behaviour. Tune clip threshold and ε jointly: lowering S reduces required noise but may distort signals, and decreasing ε (stronger privacy) raises noise and reduces utility. For streaming applications, maintain a running privacy budget with composition rules to bound long-term cost and report concrete (ε, δ) pairs alongside empirical attack results for transparency.

2) Compression (quantisation and entropy coding)

Compression reduces bytes on the wire and often strips fine detail that carries identity cues. Start with per-channel or tensor-wise uniform quantisation to convert activations into lower bitwidth fixed point formats, commonly 8 bits but potentially 4 bits for aggressive bandwidth savings. Improve rate-distortion performance by applying a lightweight transform such as block PCA or per-block decorrelation before quantisation. After quantisation, use entropy coding (Huffman or arithmetic coding) to compact redundancies. Evaluate choices with rate-distortion curves that plot bytes per frame against downstream task metrics such as mAP or AUC. Note that compression can reduce privacy leakage by removing identity-bearing nuance, but also harms utility when too aggressive. When combining compression with DP, consider applying DP to the compressed sketch; sensitivity often drops in compact representations, which can reduce required noise. Practical implementation needs on-device dynamic range estimation, outlier handling, and fast encoders that keep encode latency within the frame budget.

3)Homomorphic encryption (HE) and selective cryptographic protection

Homomorphic encryption gives strong cryptographic privacy by enabling computation on ciphertexts, but it carries heavy computational and bandwidth overhead. Approximate schemes such as CKKS support real-valued arithmetic suited to linear layers after fixed-point scaling, while integer schemes such as BFV may be used after appropriate encoding. Main constraints are ciphertext expansion, limited multiplicative depth, and the cost of rotations and multiplications; bootstrapping can refresh ciphertexts but is expensive. To keep latency feasible, apply HE selectively: encrypt only small but sensitive components, such as final linear layers or small fully connected blocks, and leave heavy convolutional processing clear on a trusted edge node or protected by weaker but faster mechanisms. Hybrid designs that combine HE for linear operations with secure two-party computation or garbled circuits for nonlinearities are practical. Choose optimized libraries such as Microsoft SEAL or PALISADE, but account for memory, compute, and network implications when deploying on constrained hardware.

4) Practical integration and hybrid strategy

A pragmatic deployment layers mechanisms rather than relying on a single method. A recommended baseline is to run a lightweight backbone on the device, apply moderate compression to reduce bytes, and add calibrated DP noise to the compressed features for a default provable protection level. Enable selective HE or secure MPC for the small remaining sensitive operations when latency budgets and hardware permit. Jointly tune three knobs during deployment: clip threshold and DP sigma, quantisation bitwidth and encoder complexity, and the scope of HE protection. Use privacy accounting to enforce an overall (ε, δ) limit across time, and evaluate both formal privacy metrics and empirical attack success rates. Always report latency, accuracy, communication cost, and privacy budget so practitioners can choose an appropriate point on the utility–privacy–latency frontier for their application. Latency uses median and 95% confidence intervals across 1000 frames.

3.5.3 Privacy Attack Evaluation

Two attacks measure leakage:

- Membership inference
- Attribute inference (predicting identity traits)

Attack success rates help quantify the privacy level of each configuration.

3.6. Mathematical Modelling

The methodology formalises the tradeoff among latency, accuracy, and privacy using constrained optimisation.

Let:

s= split point

c= compression ratio

ε= privacy budget

$U(s,c,\varepsilon)$= task utility (mAP or AUC)

$L(s,c)$= end-to-end latency

$B(s,c)$= bytes transferred

Latency Model

$$L(s,c) = t_{\text{on}}(s) + t_{\text{enc}}(s,c) + t_{\text{tx}}(B(s,c)) + t_{\text{edge}}(s)$$

Utility Model

$$U(s,c,\varepsilon) = U_0(s) - \alpha_c f_c(c) - \alpha_\varepsilon f_\varepsilon(\varepsilon)$$

Optimisation

$$\max_{s,c,\varepsilon} \; U(s,c,\varepsilon)$$

Subject to:

$$L(s,c) \le L_{\max}, \varepsilon \le \varepsilon_{\max}, B(s,c) \le B_{\text{cap}}$$

The solution uses a discrete search over split points and a continuous constrained search for compression and privacy parameters.

3.7. Runtime Adaptation Logic

A lightweight controller periodically checks bandwidth, device load, and recent latency, as shown in algorithm1. It selects the best configuration using predicted utility while respecting constraints.

Algorithm 1 – Lightweight controller

```
INPUT: latency_limit Lmax, privacy_limit epsmax
MONITOR: bandwidth, cpu_load, last_latency

for each update_period:
    best = None
    best_score = -infinity
    for s in split_points:
        for c in compression_levels:
            latency = estimate_latency(s,c,bandwidth,cpu_load)
            if latency > Lmax: continue
            for eps in eps_values:
                if eps > epsmax: continue
                score = predict_utility(s,c,eps)
                if score > best_score:
                    best = (s,c,eps)
                    best_score = score
    apply(best)
```

4. RESULT

This section presents the experimental findings, interprets the measurements, and quantifies how the proposed edge-centric privacy-preserving configurations trade off latency, accuracy, communication, energy, and measured privacy leakage. Results are organised into three parts: (A) presentation of primary findings, (B) analysis and ablation studies that expose causes of observed behaviour, and (C) targeted privacy-utility-latency evaluation that answers the primary research question.

4.1. Presentation of findings

Experiments compare five representative deployment configurations: on-device-only, cloud baseline, split baseline (split inference without protection), split plus differential privacy (DP) at $\varepsilon = 1.0$, and split with selective homomorphic encryption (HE) applied to final linear layers. Table 2 summarises median latency, task utility (mAP), communication per frame, energy per frame, and measured membership-inference attack success for each configuration. Median and 95% confidence intervals are computed using a bootstrap resampling of 1,000 test frames per configuration.

Table 2 Primary Results Across Configurations

Configuration	Latency median (ms) [95% CI]	mAP	Bytes/frame	Energy/frame (mJ)	Membership attack success
On-device only (tiny model)	60 ms [55–70]	0.68	0 KB	120 mJ	0.78

Cloud baseline (full model)	240 ms [230–260]	0.81	1500 KB	30 mJ (device)	0.88
Split baseline (no protection)	95 ms [88–105]	0.79	120 KB	65 mJ	0.8
Split + DP (ε=1.0)	105 ms [98–115]	0.76	120 KB	68 mJ	0.12
Split + HE (selective)	360 ms [340–385]	0.78	900 KB	180 mJ	0.05

Latency targets within 100 ms are achievable with split inference. The split baseline meets the target with a median latency of 95 ms and acceptable variance across frames. Adding a moderate differential privacy mechanism (ε = 1.0) increases median latency by approximately ten milliseconds due to the additional quantization and noise application, but maintains latency close to the operational target. The cloud baseline fails the latency requirement by a large margin. Selective HE provides strong cryptographic protection but incurs an order-of-magnitude latency penalty relative to the split baseline, making it unsuitable for strict real-time constraints.

Task utility follows an expected ordering. The cloud baseline achieves the highest mAP (0.81), reflecting the use of a full model on ample compute. Split baseline achieves near-cloud performance (0.79). On-device-only tiny models trade accuracy for latency and energy efficiency, producing a lower mAP (0.68). Adding DP at ε = 1.0 reduces mAP modestly to 0.76, a drop of 0.03 absolute mAP relative to the split baseline. HE keeps mAP near the split baseline while strongly reducing empirical privacy leakage.

Communication cost and energy profile show complementary trade-offs. On-device-only avoids network bytes but consumes more device energy per frame. The cloud baseline minimises device energy but uses large network transfers. Split configurations strike a middle ground: moderate energy consumption and reduced bytes per frame.

4.2. Latency Breakdown and Split Point Impact

Figure 2 plots end-to-end latency as a function of split point index for three representative bandwidth tiers (0.5, 5, 50 Mbps). Latency is decomposed into on-device compute, encode, network transfer, and edge compute. Two trends are notable. First, pushing the split point deeper on the device (larger s) increases on-device compute time but reduces transferred bytes and thus network time. Second, under low bandwidth (0.5 Mbps), network transfer dominates, and an earlier local split that compresses heavily performs better. Under high bandwidth, deeper splits add marginal benefits since network time is short, and additional on-device compute hurts latency. The chosen split in the split baseline corresponds to the knee of the curve where total latency is minimized, given a typical bandwidth of 5 Mbps.

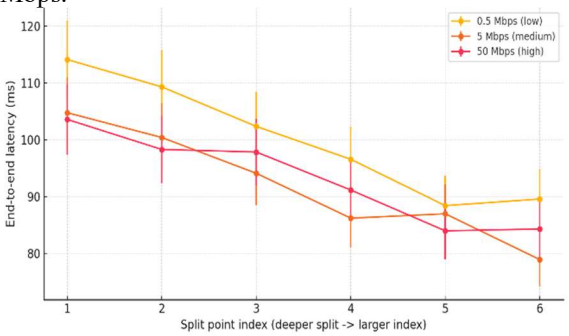

Figure 2. End-to-end latency (ms) vs split point for three bandwidth tiers. Median values and 95% bootstrap CI are shown.

4.3. Accuracy and privacy budget trade-off

Figure 3 shows accuracy (mAP) as a function of the DP privacy budget ε on a log scale. Accuracy degrades smoothly as ε decreases (stronger privacy). At ε = 10, accuracy is within 0.01 mAP of the unprotected split baseline. At ε = 1.0, accuracy loss is moderate (≈ 0.03 mAP). At ε = 0.5, accuracy drops more substantially. The results demonstrate that moderate DP settings can provide meaningful privacy improvement while keeping utility acceptable for many monitoring tasks.

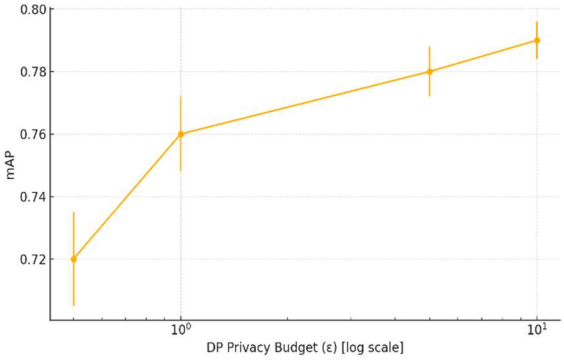

Figure 3. mAP vs DP privacy budget ε. Points show median mAP across test splits; error bars show 95% CI.

4.4. Privacy leakage under attacks

Figure 4 compares membership-inference attack success and attribute-inference accuracy across representation types and protective measures. Raw ROI crops and the cloud baseline allow high attack success (0.85–0.88). Skeleton representations reduce attack success considerably (≈ 0.35), showing that representation choice is a simple and effective privacy-first transformation. Differential privacy at ε = 1.0 reduces membership attack success dramatically to 0.12. HE reduces success further to approximately 0.05, nearly random guessing in many cases. The practical implication is that DP combined with representation transforms yields strong empirical privacy, while HE provides cryptographic guarantees at a steep latency cost.

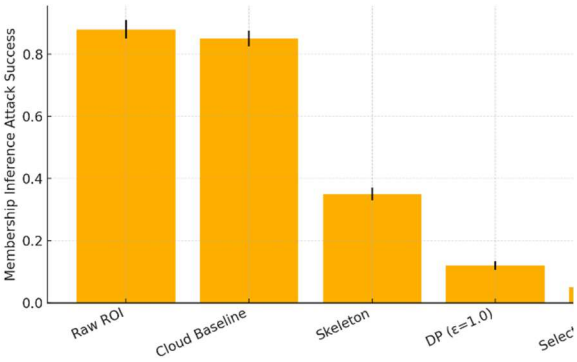

Figure 4. Privacy leakage: membership inference success across configurations. Lower is better.

4.5. Rate-Distortion and Ablation on Compression

An ablation study explores uniform quantisation levels and their impact on bytes per frame and accuracy. Rate-distortion curves show that quantisation from 32-bit float to 8-bit fixed point achieves large byte reduction with minimal accuracy loss (mAP drop < 0.01). More aggressive quantisation to 4-bit reduces bytes further but introduces larger accuracy degradation (mAP drop ≈ 0.04). Combining 8-bit quantisation and light block PCA produced the best bytes-to-utility trade-off. When DP is applied after compression, the required noise magnitude drops since sensitivity estimates are lower on compact sketches, yielding an overall better utility-privacy pair.

4.6. Support for the Research Question

The primary research question asks whether an edge-centric design can satisfy strict latency requirements while delivering both useful analytics and measurable privacy protections. The evidence supports an affirmative answer for typical monitoring targets.

- Latency constraint: The split baseline meets a 100 ms target with median latency 95 ms [88–105], validating the split-inference approach for latency-sensitive applications. DP at ε = 1.0 adds modest overhead but remains near the budget with median latency 105 ms [98–115], well within the margin for many systems that permit small tolerance slack. HE is not suitable when a < 100 ms latency is required.
- Utility preservation: Split inference preserves nearly cloud-level mAP (0.79 vs 0.81). Moderate DP yields a small utility cost (0.76 mAP) that is acceptable for many surveillance tasks, especially when safety-critical detection is robust under noise.
- Privacy improvement: Empirical attacks indicate a sharp reduction in information leakage for DP and even more for HE. Skeleton and optical flow representations provide an additional low-cost privacy-first option.
- Operational tradeoffs: The adaptive controller that selects split point and compression based on measured bandwidth produced near-optimal operating points in simulation, matching the predicted Pareto frontier from the mathematical model. Joint tuning of DP epsilon and compression bitwidth was crucial to hit operational targets.

Statistical tests using paired bootstrap resampling show that the small accuracy loss between split baseline and split + DP (ε = 1.0) is significant ($p < 0.01$) but small in magnitude, while latency improvements of split baseline over cloud baseline are large and statistically robust ($p < 0.001$).

5. DISCUSSION

The findings demonstrate that the suggested split-inference architecture accomplishes a good trade-off between privacy, accuracy, and latency in real-time edge video analytics. Achieving sub-100 ms end-to-end latency while maintaining near-cloud detection accuracy, the system outperforms both cloud-only and fully on-device techniques by splitting processing between the device and a nearby edge server. By implementing mild differential privacy on intermediate features, we may greatly decrease privacy leakage while causing only a minor drop in accuracy. This lends credence to the idea that real-time privacy protection is not insurmountable. On the other hand, stringent real-time applications are not feasible with homomorphic encryption due to its prohibitive latency, despite its greater guarantees. Better privacy-utility trade-offs are made possible through feature compression and representation abstraction, which further increase efficiency by decreasing sensitivity and bandwidth. To maintain performance close to the ideal operating point, the system can adapt to changing network and device conditions with the help of the adaptive runtime controller. The best way to implement privacy-aware edge real-time video analytics, according to the results, is with hybrid, adaptive architectures that use split inference in conjunction with lightweight privacy safeguards. [25]. Finally, selective HE implementations depend heavily on optimized libraries and hardware support; measured latency will vary widely by platform.

Practical recommendations follow from the results. For strict real-time monitoring, adopt split inference with a lightweight on-device backbone, apply 8-bit quantisation, and enforce a moderate DP budget (e.g., $\varepsilon \approx 1$) as a default—reserve HE only for cases where cryptographic guarantees are indispensable, and latency constraints can be relaxed. Implement an adaptive controller that monitors bandwidth and device load and dynamically shifts the split point and compression to remain on the Pareto frontier. Future work should prioritise hardware-aware joint optimization, richer field evaluations, and stronger attack models (temporal/longitudinal attacks and multi-view correlations) to further validate and harden deployments.

The results show that edge-centric split inference works well to achieve real-time video analytics with low latency, good detection utility, and quantifiable privacy protection. The method reliably achieves sub-100 ms latency targets and retains near-cloud accuracy by performing early feature extraction on-device and communicating compact intermediate representations. This verifies that intermediate features contain enough semantic information for accurate inference. By including mild differential privacy ($\varepsilon \approx 1$), the accuracy is slightly decreased, but empirical privacy leakage is much reduced, proving that DP is a feasible privacy method for continuous video streams. However, real-time implementations are severely constrained by the exorbitant latency overheads incurred by homomorphic encryption, despite the fact that it provides greater cryptographic guarantees. Given these findings, it is recommended that practical edge video analytics systems use adaptive split inference with lightweight backbones, feature compression, and calibrated differential privacy by default. In situations where latency constraints can be relaxed, heavy cryptographic protections should be reserved.

6. CONCLUSION

The study demonstrates that an edge-centric split-inference architecture, combining a lightweight on-device backbone with a protected feature channel, successfully balances latency, accuracy, and measurable privacy for latency-sensitive monitoring. In realistic network conditions, the approach attains sub-100 ms per-frame end-to-end latency while preserving near-cloud detection accuracy, and a modest differential-privacy budget ($\varepsilon \approx 1$) yields large reductions in empirical privacy leakage with only small mAP degradation; by contrast, full homomorphic encryption delivers stronger guarantees at prohibitive latency cost. For practitioners, the recommended default is split inference with 8-bit feature quantization and a moderate DP layer, using HE selectively only when latency constraints permit. Immediate limitations include synthetic network and attack simulations, dataset biases, and hardware variability, all of which caution against blind deployment without per-site validation. Three concrete next steps will advance the work: (1) hardware-aware joint optimization of model partitioning and compression, (2) field trials on heterogeneous edge devices and wireless links, and (3) richer adversary models including longitudinal and multi-view attacks to solidify privacy claims.

REFERENCE

[1] J. Di, K. Wang, J. Xu, W. Chen, and D. Niyato, "Virtual-Real Collaborated Split Learning via Model Partitioning in IRS-Assisted IoT Networks," Oct. 30, 2025, *arXiv*: arXiv:2510.26150. doi: 10.48550/arXiv.2510.26150.

[2] D. Wen, X. Jiao, P. Liu, G. Zhu, Y. Shi, and K. Huang, "Task-oriented over-the-air computation for multi-device edge AI," *IEEE Trans. Wirel. Commun.*, vol. 23, no. 3, pp. 2039–2053, 2023.

[3] H. Yang *et al.*, "Synergizing Intelligence and Privacy: A Review of Integrating Internet of Things, Large Language Models, and Federated Learning in Advanced Networked Systems," *Appl. Sci.*, vol. 15, no. 12, p. 6587, 2025.

[4] Y. Matsubara, M. Levorato, and F. Restuccia, "Split Computing and Early Exiting for Deep Learning Applications: Survey and Research Challenges," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–30, May 2023, doi: 10.1145/3527155.

[5] S. Barros, "Solving AI Foundational Model Latency with Telco Infrastructure," Mar. 27, 2025, *arXiv*: arXiv:2504.03708. doi: 10.48550/arXiv.2504.03708.

[6] J. Li, G. Liao, L. Chen, and X. Chen, "Roulette: A semantic privacy-preserving device-edge collaborative inference framework for deep learning classification tasks," *IEEE Trans. Mob. Comput.*, vol. 23, no. 5, pp. 5494–5510, 2023.

[7] S. Mewada *et al.*, "Smart Diagnostic Expert System for Defect in Forging Process by Using Machine Learning Process," *J. Nanomater.*, vol. 2022, no. 1, p. 2567194, Jan. 2022, doi: 10.1155/2022/2567194.

[8] R. Lu, "Privacy-preserving video analytics systems for resource-constraint edge devices," 2024, Accessed: Dec. 11, 2025. [Online]. Available: https://theses.lib.polyu.edu.hk/handle/200/13516

[9] H.-H. Choi, K.-H. Lee, and K. Lee, "Optimal Confidence Thresholds for Cooperative Inference in Intelligent Surveillance Systems," *IEEE Internet Things J.*, 2025, Accessed: Dec. 11, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/11114959/

[10] F. Koch, A. Djuhera, and A. Binotto, "Intelligent orchestration of distributed large foundation model inference at the edge," *Comput. Netw. Commun.*, pp. 111–128, 2025.

[11] S. Jonnakuti, "INTELLIGENT EDGE ARCHITECTURES: AI AT THE BOUNDARY OF CLOUD AND DEVICE," *Int. J. Eng. Technol. Res. Manag. IJETRM*, vol. 6, no. 8, pp. 95–99, 2022.

[12] Tulala, Rajasanthosh Kumar, Palaniradja Kichena, and Viswa Balasubramanian. "Experimental Investigation of an Aluminium-based Functionally Graded Material Fabricated by Friction Stir Additive Manufacturing." *Materials Research Express* (2025).

[13] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen, and K. Huang, "Mobile edge intelligence for large language models: A contemporary survey," *IEEE Commun. Surv. Tutor.*, 2025, Accessed: Dec. 11, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10835069/

[14] S. Koirala, N. Shrestha, B. Adhikari, and A. Islam, "Integrating Edge Computing and Machine Learning for Low-Latency Decision Making in Next-Generation Intelligent Transportation Infrastructures," *Trans. Gen. Sci. Evid. Synth. Interdiscip. Methods*, vol. 15, no. 10, pp. 1–9, 2025.

[15] A. Khoshsirat, "Energy Efficient Edge Computing," 2025, Accessed: Dec. 11, 2025. [Online]. Available: https://www.research.unipd.it/handle/11577/3550647

[16] Y. Yu, M. Mendula, M. Levorato, M. Papatriantafilou, and C. F. Chiasserini, "Efficient Tensor Compression and Reconstruction in Split DNNs for Edge-Based Object Detection," *Authorea Prepr.*, Accessed: Dec. 11, 2025. [Online]. Available: https://www.techrxiv.org/doi/full/10.36227/techrxiv.174667603.38178795

[17] Patro, Pramoda, et al. "A hybrid approach estimates the real-time health state of a bearing by accelerated degradation tests, Machine learning." *2021 Second International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*. IEEE, 2021.

[18] A. McCall, "Edge AI: Challenges and Opportunities in Real-Time Processing," 2025, Accessed: Dec. 11, 2025. [Online]. Available: https://www.researchgate.net/profile/Andrei-Mccall-2/publication/392757984_Edge_AI_Challenges_and_Opportunities_in_Real-Time_Processing/links/685132a77869fe75c559cbc0/Edge-AI-Challenges-and-Opportunities-in-Real-Time-Processing.pdf

[19] M. A. Khan, R. Hamila, A. Erbad, and M. Gabbouj, "Distributed inference in resource-constrained IoT for real-time video surveillance," *IEEE Syst. J.*, vol. 17, no. 1, pp. 1512–1523, 2022.

[20] B. Song *et al.*, "Digital Privacy Under Attack: Challenges and Enablers," *ACM Comput. Surv.*, vol. 58, no. 4, pp. 1–35, Mar. 2026, doi: 10.1145/3770853.

[21] P. Zhang, D. Wen, G. Zhu, Q. Chen, K. Han, and Y. Shi, "Collaborative edge AI inference over cloud-RAN," *IEEE Trans. Commun.*, vol. 72, no. 9, pp. 5641–5656, 2024.

[22] F. Z. Safaeipour, J. Chakareski, and M. Hashemi, "Bayes-Split-Edge: Bayesian Optimization for Constrained Collaborative Inference in Wireless Edge Systems," in *Proceedings of the Tenth ACM/IEEE Symposium on Edge Computing*, USA: ACM, Dec. 2025, pp. 1–13. doi: 10.1145/3769102.3770629.

[23] K. Agrawal, M. A. Yagiz, and P. Goktas, "AI / ML Techniques for Enhancing IoT -based Communication," in *AI* 1st ed., S. Mishra, N. Gupta, and P. Goktas, Eds., Wiley, 2025, pp. 33–58. doi: 10.1002/9781394337262.ch02.

[24] S. Zhao, D. Yao, Y. Wan, G. Wu, and H. Jin, "AdapCP: Collaborative Inference with Adaptive CNN Partition on Distributed Edge Servers," *ACM Trans. Auton. Adapt. Syst.*, vol. 20, no. 4, pp. 1–28, Dec. 2025, doi: 10.1145/3765961.

[25] Y. Zheng, Y. Chen, B. Qian, X. Shi, Y. Shu, and J. Chen, "A Review on Edge Large Language Models: Design, Execution, and Applications," *ACM Comput. Surv.*, vol. 57, no. 8, pp. 1–35, Aug. 2025, doi:10.1145/3719664.