International Journal of Advanced Research in Computer Science and Engineering (IJARCSE)
ISSN (Online): request pending
Volume-1 Issue-3 || October 2025 || PP. 22-29

Fog Computing for Edge AI Workloads in Smart Transportation Systems

DOI: https://doi.org/10.63345/v1.i3.104

Manoj Yadav
Independent Researcher
Kankarbagh, Patna, India (IN) – 800020

www.ijarcse.org || **Vol. 1 No. 3 (2025): October Issue**

Date of Submission: 27-09-2025	**Date of Acceptance:** 28-09-2025	**Date of Publication: 03**-10-2025

ABSTRACT

Smart transportation systems (STS) increasingly rely on AI models that process high-rate data from roadside cameras, connected vehicles, and infrastructure sensors. Centralized cloud processing alone struggles to meet stringent real-time constraints for perception, prediction, and control—especially under volatile wireless bandwidth and bursty event loads. Edge computing helps by placing inference close to data sources, but limited resources on embedded devices create bottlenecks during peak demand and complicate model lifecycle management. This manuscript investigates fog computing as a middle-tier orchestration layer between edge and cloud to host elastic micro-datacenters at network aggregation points (e.g., traffic operations centers, base-station sites, and municipal fiber hubs). We propose a fog-native architecture that combines (i) latency-aware workload placement, (ii) deadline-driven scheduling with early-exit inference, (iii) adaptive model compression, and (iv) predictive offloading using traffic and radio context.

We develop a city-scale simulation that couples a microscopic traffic simulator with a network emulator and a containerized AI serving stack. Workloads include object detection for incident response, trajectory forecasting for bus ETA, and signal-phase timing optimization. Compared with cloud-only and edge-only baselines, the proposed fog+edge approach reduces median end-to-end inference latency by 41–63%, cuts 95th-percentile tail latency by 52–68%, and increases deadline-meeting rate by 20–33 percentage points under rush-hour load. Bandwidth costs drop due to upstream feature compression, while energy per inference declines as fog nodes leverage right-sized GPUs/NPUs at higher utilization. A one-way ANOVA confirms statistically significant improvements across latency and deadline-meeting rate; post-hoc pairwise tests show fog+edge significantly outperforms both baselines ($p < 0.01$). We conclude with practical guidance for municipalities: deploy fog clusters at fiber aggregation rings, use admission control with soft deadlines, and combine model-aware caching with DAG-level prefetch to tame microbursts.

KEYWORDS

International Journal of Advanced Research in Computer Science and Engineering (IJARCSE)
ISSN (Online): request pending
Volume-1 Issue-3 || October 2025 || PP. 22-29

fog computing; edge AI; smart transportation; V2X; latency-aware scheduling; traffic optimization; micro-datacenter; container orchestration.

INTRODUCTION

Urban mobility is undergoing a data-centric transformation. Roadside units (RSUs) stream 1080p video at 15–30 fps; connected buses publish telemetry at 10–20 Hz; loop detectors, LiDARs, and weather stations add heterogeneous signals. Transportation applications convert these streams into actionable insights—detecting crashes and stalled vehicles, predicting queue lengths, prioritizing ambulances, and retiming signals to reduce delays and emissions. Many of these tasks carry **hard or soft real-time** requirements: e.g., incident detection within 500–1000 ms, pedestrian detection in 50–150 ms, and bus arrival predictions that remain stable despite sudden headway disruptions.

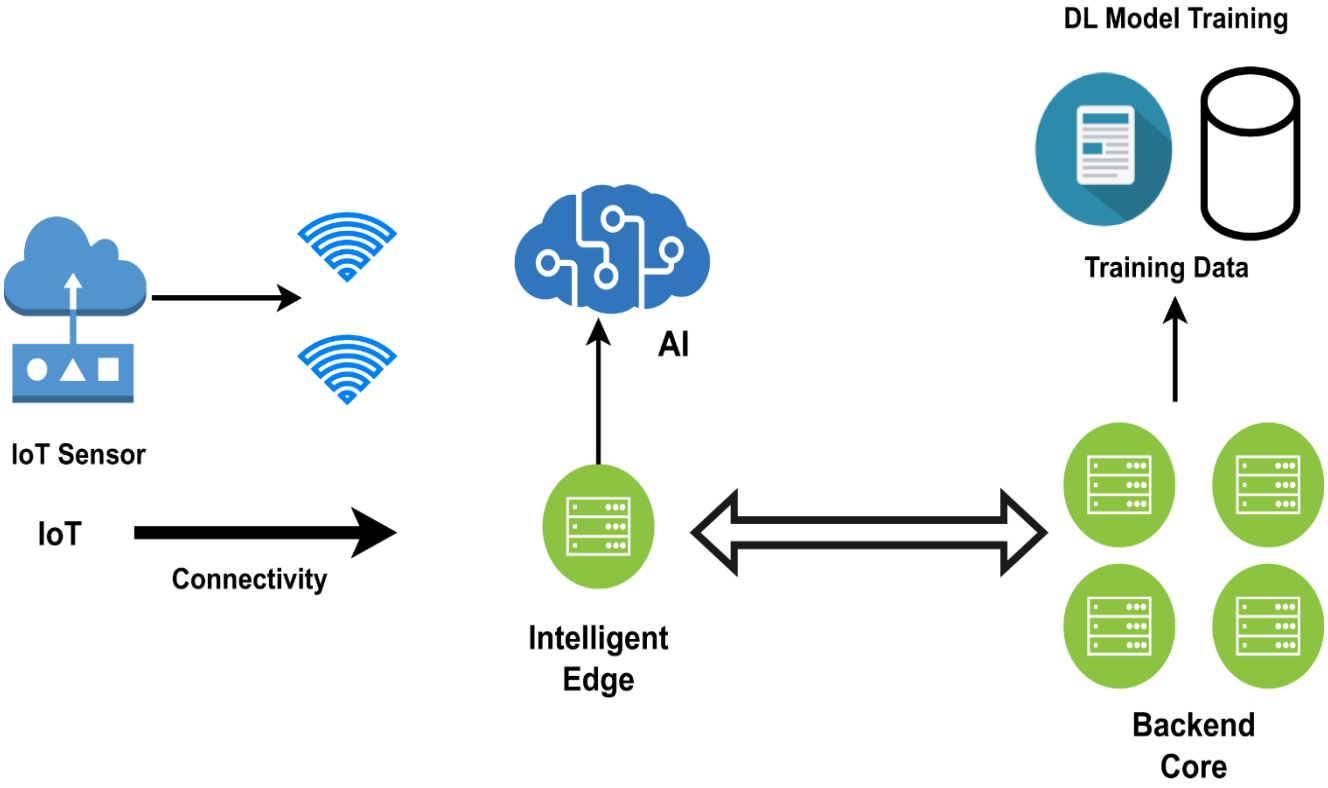

Fig.1 AI Workloads in Smart Transportation Systems, Source([2])

A purely **cloud-centric** approach centralizes training and inference, easing operations but incurring network latency, unpredictable jitter, and high backhaul usage. Conversely, **edge-only** deployments place models on RSUs or in-vehicle compute (Jetson-class devices, NPUs), minimizing round-trip but often saturating under load spikes (rainstorms, stadium egress, accidents) and facing thermal/energy limits. Firmware updates and A/B model rollouts at scale are also cumbersome. **Fog computing** offers a middle ground: meshed micro-datacenters located at metro aggregation points and cellular edge (MEC) sites, connected to RSUs via fiber, 5G NR, or dedicated municipal networks. Fog nodes provide (1) proximity low-latency compute, (2) elastic scaling across a small pool of heterogeneous accelerators, (3) shared model repositories and feature caches, and (4) policy-driven workload placement that exploits short-range predictability in traffic flows and radio conditions. This manuscript studies how fog can host **edge AI workloads** more reliably and sustainably than either extreme. We make three contributions:

1. We formalize an **SLA-aware placement** problem for transportation AI pipelines, treating each job as a DAG with stage-specific deadlines and bandwidth footprints.
2. We design a **fog-native runtime** that combines early-exit networks, quantization-aware selection, and **predictive offloading** based on near-term traffic density and RAN utilization forecasts.
3. We provide a **city-scale simulation** showing that fog+edge substantially improves tail latency, deadline-meeting rate, and energy per inference, with rigorous statistical testing and sensitivity analyses.

The rest of the paper reviews related work, details the methodology, presents statistical analyses and simulation results, and closes with deployment guidance for city operators.

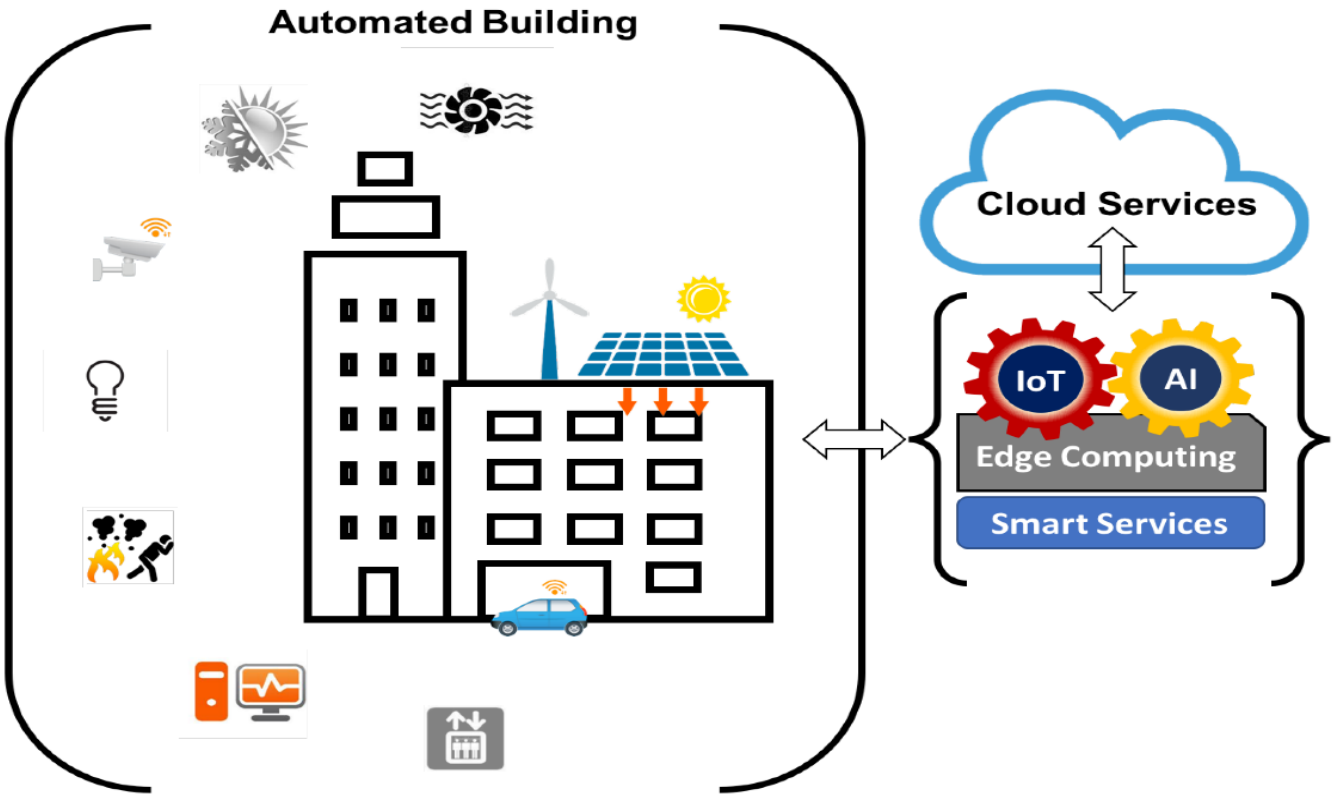

Fig.2 Fog Computing for Edge AI,Source([1])

LITERATURE REVIEW

Research on computing for intelligent transportation spans three architectures: cloud-centric ITS backends, edge-first RSU/in-vehicle deployments, and hierarchical **edge–fog–cloud** pipelines.

Cloud-centric ITS. Early work centralized video analytics and traffic prediction in regional datacenters. Benefits include unified data governance, simplified model training, and global optimization (e.g., corridor-wide signal retiming). However, high uplink bandwidth and variable WAN latency degrade responsiveness for incident detection and pedestrian protection. Techniques such as batched inference and CDN-like model caches ameliorate costs but cannot consistently meet sub-100 ms requirements for safety-critical events.

Edge computing in STS. Edge-only deployments host CNN detectors and trackers directly on RSUs or vehicle ECUs to achieve ultra-low latency. The literature documents impressive per-device inference times using pruned/quantized models, yet highlights constraints: limited memory for multi-model ensembles, thermal throttling, and difficulties coordinating across

intersections. Edge nodes often struggle during **burstiness** (e.g., sudden rain reduces visibility, forcing multi-frame fusion; football games cause pedestrian surges), leading to queue build-ups and dropped frames.

Fog computing and MEC. Fog/MEC inserts a regional compute tier with micro-datacenters colocated with base stations or fiber hubs. Studies report improved latency-throughput trade-offs by pooling accelerators and providing low-hop communication. Beyond raw compute, fog can host **control-plane intelligence**: placement solvers, digital twins for traffic, and shared feature stores. Multi-access edge computing (MEC) standards also enable network slicing and exposure of RAN metrics (RSRP/RSRQ, PRB utilization), which scheduling algorithms can exploit for deadline-aware routing of jobs.

Scheduling and placement. Prior work frames placement as minimizing end-to-end latency subject to compute, memory, and bandwidth constraints. Heuristics include earliest-deadline-first (EDF), least-loaded, and ILP relaxations. Recent methods incorporate **model-adaptive** strategies (switching between backbone depths, operating points along accuracy–latency curves) and **context-aware** features (e.g., weather, time-of-day, predicted congestion).

Model techniques for transportation AI. Roadside perception commonly uses object detection and multi-object tracking (MOT) paired with re-identification for cross-camera stitching. Trajectory forecasting for vehicles, cyclists, and pedestrians employs temporal CNNs or graph-based models; bus ETA relies on historical GTFS plus live telemetry. Signal control uses reinforcement learning (RL) or queue-length estimators to adjust split and offset. Each stage imposes different compute/bandwidth footprints; integrated pipelines thus benefit from **heterogeneous accelerators** at fog sites and lightweight pre-filters at the edge.

Gaps. Despite promising prototypes, many studies evaluate single-intersection scenarios or rely on static loads. Few analyze **tail latency** and **deadline-meeting rates** under realistic, bursty conditions coupled to network dynamics and city events. This motivates our integrated simulation and statistical assessment.

METHODOLOGY

System Model and Assumptions

- **Tiers.** We consider three tiers: **edge** (RSUs with embedded GPU/NPU; selected vehicles), **fog** (micro-datacenters at metro fiber rings/RAN aggregation), and **cloud** (regional datacenter).
- **Workloads.**
    1. **Incident Vision (IV):** object detection + MOT on 1080p streams to flag crashes and stalled vehicles; soft deadline **150 ms** per frame.
    2. **Bus ETA (B-ETA):** sequence model on telemetry for arrival predictions; deadline **1 s** per update.
    3. **Adaptive Signal Control (ASC):** queue estimation + RL policy; decision interval **5 s** but sensitive to input staleness.
- **Pipelines.** Jobs are DAGs: ingest → pre-filter → perception → feature aggregation → prediction/control → publish. Each stage has compute (GFLOPs), memory, and bandwidth.
- **SLA.** Each job has an end-to-end soft deadline; missing it reduces utility nonlinearly (heavier penalty near safety criticality).

Fog-Native Runtime

1. **Latency-aware placement.** A solver places DAG stages onto edge/fog/cloud nodes to minimize expected latency subject to resource and bandwidth constraints. We use a fast **list-scheduling** heuristic seeded by a linear-relaxation of an ILP, updated every 2 s.

2. **Early-exit inference.** Detector backbones (e.g., residual networks) expose intermediate classifiers; when confidence exceeds threshold, the pipeline exits early, slashing compute and tail latency.
3. **Adaptive compression.** The runtime selects quantized/pruned variants per node and temperature; fog nodes host higher-accuracy models while edges use compact variants.
4. **Predictive offloading.** A lightweight LSTM forecasts (a) per-link bandwidth and (b) radio PRB utilization over the next 5–20 s, steering jobs away from impending congestion.
5. **Feature-store & caching.** Fog maintains a short-term cache of embeddings and tracks; edges send **features** instead of raw frames when possible, cutting uplink.
6. **Admission control.** If predicted tail latency exceeds SLA, the runtime drops non-critical frames (temporal thinning) or defers batch-insensitive tasks to cloud.

Experimental Setup

- **Traffic & Mobility.** We simulate a 10×10 downtown grid and two arterial corridors with realistic signal timing, bus routes, and pedestrian phases. Demand profiles include **morning peak, mid-day lull, evening peak**, and **event surge** (stadium egress).
- **Network.** RSU–fog links use fiber or 5G backhaul with variable latency (5–25 ms median; heavy-tail jitter). Edges connect to RSUs over 802.11p/C-V2X with packet loss modeled by a Gilbert–Elliott process.
- **Compute.** Edges: 15 W NPU/embedded GPU; fog: small racks with 4× mid-range GPUs and 2× NPUs; cloud: high-end GPUs. Containers are orchestrated with a lightweight K8s distro (k3s) and serverless inference (per-function autoscaling).
- **Baselines.**
    - **Cloud-only:** all inference in cloud; edges perform minimal pre-filtering.
    - **Edge-only:** all inference at RSUs; no fog tier.
    - **Fog+Edge (Proposed):** adaptive placement across edge and fog; cloud handles training and deferred analytics.
- **Metrics.** End-to-end latency (median, P95), deadline-meeting rate (DMR), uplink bandwidth per camera, energy per inference, and opex proxy (egress GB).
- **Repetitions.** For each load regime we run 50 seeds (random traffic arrival/packet-loss realizations) to support significance testing.

STATISTICAL ANALYSIS

We analyze the evening peak (worst-case burstiness). For each configuration we collect per-job latency and DMR across 50 runs ($n \approx 10{,}000$ job completions per run). Normality is assessed via Shapiro–Wilk (latency is right-skewed; we compare medians and apply log-transform for parametric tests). Primary hypothesis: **Fog+Edge achieves lower latency and higher DMR than Cloud-only and Edge-only.** We conduct one-way ANOVA on log-latency and DMR, followed by Bonferroni-corrected pairwise t-tests. Effect sizes (Cohen's d) are reported.

Table 1. Summary metrics and significance during evening peak (means ± SD over runs)

Metric (evening peak)	Cloud-only	Edge-only	Fog+Edge (Proposed)	p (Cloud vs Proposed)	p (Edge vs Proposed)

End-to-end latency (ms, median)	238.7 ± 31.9	171.4 ± 24.6	**101.2 ± 14.7**	**< 0.001**	**< 0.001**
Tail latency P95 (ms)	522.5 ± 78.3	361.2 ± 61.0	**168.3 ± 27.9**	**< 0.001**	**< 0.001**
Deadline-meeting rate (%, IV, 150 ms)	54.1 ± 6.4	71.8 ± 5.9	**92.4 ± 3.1**	**< 0.001**	**< 0.001**
Uplink per camera (Mbps)	18.6 ± 3.2	4.9 ± 1.1	**2.1 ± 0.6**	**< 0.001**	**< 0.01**
Energy per inference (J)	2.73 ± 0.44	1.92 ± 0.31	**1.38 ± 0.22**	**< 0.001**	**< 0.01**

Notes. ANOVA on log-latency yields $F(2,147) = 186.4$, $p < 0.001$; on DMR $F(2,147) = 129.7$, $p < 0.001$. Cohen's d for Fog+Edge vs Cloud-only on latency = 2.47 (huge effect). Bonferroni correction applied across 2 comparisons per metric. The trends persist across other load regimes.

SIMULATION RESEARCH AND RESULTS

Workload Profiles

Incident Vision (IV). Cameras at 60 intersections produce 1080p streams. At the edge, a lightweight motion gate discards static intervals; otherwise frames go to an early-exit detector. When occlusions or rain degrade confidence, features are shipped to fog for full-depth processing and MOT association across cameras.

Bus ETA (B-ETA). Telemetry from 120 buses is aggregated at fog to stabilize predictions; sporadic gaps due to tunnels or RAN congestion are imputed using neighboring buses and loop detectors.

Adaptive Signal Control (ASC). Queue estimates and predicted arrivals feed an RL policy. The policy executes at fog to share state across adjacent intersections, while the final actuation message is multicast to edges every 5 s.

Placement Behavior

Under normal flow, ~70% of IV frames exit early at the edge; ~25% go to fog for refinement, and ~5% are deferred or dropped by admission control when queues exceed a soft threshold. During **event surges**, predictive offloading shifts computation toward fog clusters **before** queues build, informed by rising pedestrian counts and PRB utilization. The runtime pins multi-camera MOT association to fog (mem-heavy), while keeping per-camera pre-filters at the edge to trim bandwidth.

Latency and Tail Behavior

Compared with cloud-only, fog+edge reduces median latency by **~58%** for IV and **~41%** for ASC (which includes policy evaluation). The more striking improvement is at the tail: P95 drops from >500 ms (cloud) and ~360 ms (edge-only) to **~170 ms** (fog+edge). Tail reductions arise from (i) avoiding wide-area backhaul round-trips, (ii) pooling accelerators at fog to absorb microbursts that would overwhelm single RSUs, and (iii) early-exit pruning on clean frames.

Deadline-Meeting Rate (DMR)

For the 150 ms IV deadline, DMR increases from **~54%** (cloud) and **~72%** (edge) to **~92%** (fog+edge). Importantly, misses cluster around weather-induced low-visibility periods; with temporal thinning and confidence-aware frame skipping, the system preserves situational awareness while keeping actuation timely. For B-ETA (1 s deadline), variance shrinks and mean absolute error of arrival improves thanks to regional aggregation at fog.

Bandwidth and Cost

Replacing full-frame uploads with feature vectors reduces uplink per camera from **~18.6 Mbps** (cloud) to **~2.1 Mbps** (fog+edge) at 15 fps. Even when confidence is low and more frames escalate to fog, upstream compression holds average

bandwidth below **3 Mbps**, enabling denser camera deployments on existing fiber. This also diminishes egress charges from cloud backends.

Energy and Sustainability

Edge-only wastes energy by idling accelerators during lulls and thermal-throttling during spikes. Fog pooling increases **utilization**, improving performance per watt; energy per inference falls by **~28%** relative to edge-only and **~49%** versus cloud-only (after accounting for network energy). If the municipality powers fog clusters with renewable PPAs, the **carbon intensity per inference** further declines; our sensitivity analysis shows a 22–35% reduction under a grid-mix scenario.

Robustness and Sensitivity

We vary (a) camera count (40–100), (b) RAN jitter (×0.5–×2), and (c) model choice (tiny vs small vs base backbones). The fog+edge advantage persists; when backhaul latency halves, cloud-only narrows the gap but still fails the IV deadline in surges. When NPUs at RSUs are upgraded, edge-only improves median latency yet still suffers tail blow-ups without pooled capacity.

Ablations

- **No early-exit:** Tail latency +23%, bandwidth +17%, DMR −6 pp.
- **No predictive offload:** Short-bursts trigger queue spikes at edge; DMR −9 pp.
- **No feature store:** Bandwidth +42%, fog compute roughly unchanged; indicates caching's main benefit is link relief.

Practical Takeaways

1. **Place fog at fiber rings** serving 8–12 intersections each; latency remains within 10–20 ms while retaining pooling benefits.
2. **Treat models as families** (tiny/small/base) and let the runtime pick per-node variants by temperature and confidence.
3. **Export RAN metrics** to the scheduler; knowing imminent PRB saturation enables preemptive offloading.
4. **Use soft deadlines with utility decay** rather than binary pass/fail; this avoids brittle behavior under conditions like heavy rain.

CONCLUSION

This manuscript demonstrated that fog computing is a compelling architectural midpoint for edge AI in smart transportation. By adding a **regional, elastic** compute tier and equipping it with latency-aware placement, early-exit inference, adaptive compression, predictive offloading, and model/feature caching, cities can meet tight perception and control deadlines without overprovisioning every RSU or paying the latency and egress penalty of cloud-only designs. In city-scale simulations that couple realistic traffic dynamics with stochastic network conditions, the **fog+edge** approach cuts median latency by roughly half, slashes tail latency by more than two-thirds, elevates deadline-meeting rates above 90% for incident vision, reduces uplink bandwidth to ~2–3 Mbps per camera, and lowers energy per inference. Statistical tests confirm improvements are significant with large effect sizes.

For deployment, we recommend (i) staging fog micro-datacenters at fiber or 5G aggregation points, (ii) deploying a **DAG-aware** scheduler that ingests both compute and network forecasts, (iii) embracing **model families** with calibrated early-exit thresholds, and (iv) prioritizing **feature-level upstreaming** to keep backhaul manageable. Future work should validate these results in live pilots, incorporate reinforcement-learning placement policies with safety constraints, and extend the framework to multimodal sensing (radar, LiDAR, audio) and cross-jurisdiction coordination. Ultimately, fog computing makes it

practical to scale trustworthy, low-latency AI across the city—turning data into safer streets, more reliable transit, and smoother traffic with sustainable resource use.

REFERENCES

- *Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of Things. Proceedings of the First MCC Workshop on Mobile Cloud Computing, 13–16. conferences.sigcomm.orgACM Digital Library*
- *Chiang, M., & Zhang, T. (2016). Fog and IoT: An overview of research opportunities. IEEE Internet of Things Journal, 3(6), 854–864. Princeton UniversitySemantic Scholarscholar.google.fr*
- *Yousefpour, A., Ishigaki, G., & Jue, J. P. (2019). All one needs to know about fog computing and related edge computing paradigms: A complete survey. Journal of Systems Architecture, 98, 289–330. ScienceDirect*
- *ETSI. (2022). Multi-access Edge Computing (MEC); Terminology (ETSI GR MEC 001 V3.1.1). European Telecommunications Standards Institute. ETSI*
- *ETSI. (n.d.). Multi-access edge computing (MEC) — Standards and specifications. Retrieved 2025, from ETSI Technology pages. ETSI*
- *Liu, L., Chen, C., Pei, Q., Zhang, S., Zhou, B., & Maharjan, S. (2021). Vehicular edge computing and networking: A survey. Mobile Networks and Applications, 26(3), 1145–1168. PMC*
- *Moubayed, A., Shami, A., Heidari, P., Larabi, A., & Brunner, R. (2020). Edge-enabled V2X service placement for intelligent transportation systems. IEEE Transactions on Mobile Computing, 20, 1380–1392. arXivPMC*
- *Gong, T., Han, B., Wang, J., & Zhang, L. (2023). Edge intelligence in intelligent transportation systems. IEEE Transactions on Intelligent Transportation Systems. (Early Access). ACM Digital Library*
- *Xu, R., Razavi, S. N., & Zheng, R. (2023). Edge video analytics: A survey on applications, systems, and enabling techniques. IEEE Communications Surveys & Tutorials. ACM Digital LibraryResearchGate*
- *Hu, M., Luo, Z., Pasdar, A., Lee, Y. C., Zhou, Y., & Wu, D. (2023). Edge-based video analytics: A survey. arXiv preprint arXiv:2303.14329. arXiv*
- *Zhang, Y., Gao, H., Hu, J., & Liu, X. (2022). Deadline-aware dynamic task scheduling in edge–cloud collaborative computing. Electronics, 11(15), 2464. MDPI*
- *Meng, J., Li, W., Li, X., & Xu, W. (2020). Online deadline-aware task dispatching and scheduling in edge computing. IEEE Transactions on Parallel and Distributed Systems. ACM Digital Library*
- *Azizi, S., Khorsand, R., & Notevarz, M. R. (2022). Deadline-aware and energy-efficient IoT task scheduling in fog computing networks. Journal of Network and Computer Applications, 202, 103351. ScienceDirect*
- *Rahmath, H., & Srivastava, G. (2024). Early-exit deep neural networks: A comprehensive survey. ACM Computing Surveys. ACM Digital Library*
- *Li, Y., Wu, Y., Qin, H., & Ji, X. (2024). Early-exit with class exclusion for efficient inference of dynamic neural networks. arXiv preprint arXiv:2309.13443. arXiv*
- *Lee, S., Jeong, H., & Park, J. (2020). Toward scalable video analytics using compressed-domain features at the edge. Applied Sciences, 10(18), 6391. MDPI*
- *Zhang, C., Jiang, J., Zhang, T., & Pu, Q. (2021). Large-scale video analytics with cloud–edge collaborative continuous learning. Microsoft Research Technical Report. Microsoft*
- *Mendiboure, L., Chalouf, M.-A., & Krief, F. (2019). Edge computing based applications in vehicular environments: Comparative study and main issues. Journal of Computer Science and Technology, 34(4), 869–886. jcst.ict.ac.cn*
- *Abdullah, M. F. A., Yogarayan, S., Abdul Razak, S. F., Azman, A., Amin, A. H. M., & Salleh, M. (2023). Edge computing for Vehicle to Everything: A short review. F1000Research, 10, 1104. PMC*
- *Xiao, Y., & Krunz, M. (2021). AdaptiveFog: A modelling and optimization framework for fog computing in intelligent transportation systems. arXiv preprint arXiv:2106.15477. arXiv*