MANTECH Publications

Journal of Converging Technologies in Intelligent
Volume 1, Issue 2, May-August 2025

Convergence of Edge Computing and Machine Learning for Real-Time Intelligent Decision-Making Systems: A Structured Review of Architectures, Optimisation Techniques, and Deployment Challenges

Arindam Sen[1], Sneha Bhattacharya[2]

ABSTRACT

Real-time decision-making systems such as autonomous vehicles, industrial control loops, and remote health monitors must convert streaming sensor data into actions within deadlines that range from a few milliseconds to a few hundred milliseconds. Cloud-centric machine learning (ML) pipelines struggle to meet such deadlines because of wide-area network delay, bandwidth cost, and exposure of raw data. Edge computing relocates computation close to the data source, but running modern ML models on resource-constrained edge nodes raises new problems of model size, energy consumption, and distributed training. This review-based study synthesises peer-reviewed literature published between 2015 and 2021 on the convergence of edge computing and ML, using a four-layer analytical framework that covers reference architecture, model optimisation, collaborative inference, and federated learning. The reported evidence shows that pruning and quantisation pipelines can shrink deep networks by 35 to 49 times, that partitioned inference can reduce average latency by about 3.1 times, and that federated averaging can cut communication rounds by 10 to 100 times. No single technique is sufficient: the dominant latency term shifts from the network to model inference as computation moves to the edge, so architecture, compression, and learning strategy must be co-designed. The paper concludes with limitations, practical applications, and priorities for future research, including standardised benchmarks, security, and on-device continual learning.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

KEYWORDS: *Edge computing, Machine learning, Edge intelligence, Real-time decision-making, Federated learning, Model compression, TinyML, Computation offloading, Internet of Things, Collaborative inference*

INTRODUCTION

Decision-making systems that act on physical processes are bounded by time. A collision-avoidance module, a protective relay in a power grid, and an arrhythmia alert each lose their value when the answer arrives late, and in the first two cases a late answer can be as harmful as a wrong one. The dominant machine learning (ML) deployment model of the past decade, in which models are trained and served from centralised data centres, places a wide-area network round trip between every sensor and every decision [1, 2]. Connected devices now generate data at volumes that are expensive to backhaul, and raw camera, audio, and wearable streams carry personal information that organisations are increasingly unwilling, or legally unable, to export [3].

Edge computing responds by placing computation and storage within one or a few network hops of the data source, on end devices, gateways, base-station servers, or micro data centres [1, 2]. Surveys have organised the field around computation offloading, resource allocation, and mobility management [4, 5]. Early edge deployments mostly executed lightweight analytics, but the ambition has shifted towards running full ML inference, and increasingly training, close to the data, a movement referred to as edge intelligence [3, 6].

Convergence, however, is not simply a matter of moving a trained model closer to the sensor. Deep neural networks demand memory, arithmetic throughput, and energy far beyond what microcontrollers and battery-powered nodes can supply [7]; edge servers share limited capacity among many tenants; wireless links fluctuate; and training data are scattered across heterogeneous nodes that cannot freely pool them [8, 9]. The problem addressed here is therefore stated as follows: which combinations of architecture, model optimisation, workload partitioning, and distributed learning allow ML-driven decisions to satisfy real-time latency, accuracy, energy, and privacy constraints simultaneously, and where do current approaches fall short? The paper reviews the relevant literature, identifies the research gap, states the objectives and review methodology, presents synthesised findings, and discusses their engineering significance, limitations, and future scope.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

LITERATURE REVIEW

The literature on edge intelligence has expanded rapidly since 2016. It is reviewed below along four strands that correspond to the analytical framework used later in this paper.

Edge Computing Foundations

Shi et al. defined edge computing as processing at the network edge, on both downstream and upstream data paths, and identified latency, bandwidth, and privacy as its principal motivations [1]. Satyanarayanan argued that the edge emerges from the tension between latency-sensitive, bandwidth-hungry applications and a centralised cloud, and highlighted trust and resilience as further drivers [2]. For intelligence specifically, Zhou et al. proposed a six-level rating of edge intelligence, ranging from pure cloud intelligence to fully on-device training and inference [3], while Deng et al. distinguished 'AI for edge', which uses ML to manage edge resources, from 'AI on edge', which runs ML workloads on edge hardware [6]. Comprehensive surveys by Wang et al. and Chen and Ran document the resulting applications and enabling techniques [10, 11].

Model Compression and Efficient Inference

Han et al. combined magnitude pruning, trained quantisation, and Huffman coding to shrink AlexNet from 240 MB to 6.9 MB and VGG-16 by a factor of 49 without loss of accuracy [12]. Jacob et al. showed that integer-arithmetic-only inference improves the accuracy–latency trade-off on mobile processors [13], and Hinton et al. introduced knowledge distillation, by which a compact student network inherits the behaviour of a larger teacher [14]. Architectural efficiency has been pursued in parallel: MobileNets replace standard convolutions with depthwise separable ones and expose width and resolution multipliers for tuning [15]. At the extreme end, MCUNet co-designs the network and the inference engine to exceed 70% ImageNet top-1 accuracy on a commercial microcontroller [16]. Sze et al. emphasised that algorithm and hardware must be co-designed because data movement, rather than arithmetic, dominates energy consumption [7].

Collaborative Inference and Computation Offloading

Where a single node is insufficient, inference can be divided. Neurosurgeon partitions a network at the layer boundary that best balances computation against the size of intermediate tensors, reporting an average latency reduction of about 3.1 times and a mobile energy saving

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

of roughly 59% relative to cloud-only execution [17]. Edgent extends this idea by pairing partitioning with early-exit ‘right-sizing’ so that accuracy is traded against a stated latency deadline [18]. Mach and Becvar classified offloading decisions by what, when, and where to offload, and by the resulting energy and delay objectives [5]. Because wireless conditions and server load are stochastic, deep reinforcement learning, popularised by Mnih et al., has become an attractive policy learner for such decisions [19].

Federated and Distributed Learning at the Edge

Training at the edge raises a different problem, since data stay where they are generated. McMahan et al. proposed federated averaging, in which devices train locally and a server averages the resulting models, reporting 10 to 100 times fewer communication rounds than synchronised stochastic gradient descent [8]. Bonawitz et al. described the system design required to run this protocol across large populations of mobile devices [20], and Li et al. identified expensive communication, systems heterogeneity, statistical heterogeneity, and privacy as the defining challenges [9]. Wang et al. derived an adaptive scheme that chooses the number of local updates between global aggregations under a resource budget [21]. Kairouz et al. and Lim et al. consolidate open problems such as non-identically distributed data, client dropout, and incentive design [22, 23]. Park et al. link these themes to wireless network intelligence, where communication and learning are optimised jointly [24].

RESEARCH GAP

Existing surveys examine compression, offloading, and federated learning largely in isolation, and most evaluate them against accuracy or throughput rather than against the deadline of a closed decision loop [10, 11, 25]. Reported gains are measured on different hardware, datasets, and network conditions, and community benchmarks for edge and tiny ML are only now emerging [26, 27], which makes cross-technique comparison difficult. What is missing is an integrated, layer-wise framework that places architecture, model optimisation, collaborative inference, and distributed learning on a common set of real-time decision metrics. This review addresses that gap.

OBJECTIVES

- To establish a layered reference architecture for real-time decision systems that integrate edge computing and ML.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

- To synthesise the quantitative evidence reported for model compression, collaborative inference, and federated learning at the edge.
- To compare cloud-centric, edge-based, and on-device deployment paradigms in terms of latency, bandwidth, privacy, and compute capacity.
- To relate the reviewed techniques to representative real-time application domains and to their engineering significance.
- To identify open challenges and prioritise directions for future research.

METHODOLOGY

Review Design And Source Selection

A structured narrative review was adopted, following the planning, selection, and extraction stages recommended for software and systems engineering reviews [28]. Peer-reviewed journal and conference papers from IEEE, ACM, Elsevier, and Springer outlets were considered together with a small number of widely cited preprints that define core techniques, covering the period 2015 to 2021. A source was retained when it (i) addressed ML inference or training on or near the network edge, (ii) reported quantitative results or a formal architecture or taxonomy, and (iii) was relevant to latency-sensitive or resource-constrained operation. Studies on purely cloud-based ML, and position papers without supporting evidence, were excluded. Each retained study was appraised for the clarity of its experimental set-up, the availability of baselines, and the transparency of its reported metrics, and results that did not state the hardware or dataset behind them were treated with caution.

Analytical Framework And Synthesis Procedure

Retained studies were organised into four layers, namely reference architecture, model optimisation, collaborative inference and offloading, and federated or distributed learning, as shown in Figure 1, which separates what runs on the device, what runs at the edge, and what remains in the cloud, so that every reviewed technique can be located at a specific tier and linked to the data flows it generates. Each study was then characterised against the six evaluation dimensions defined in Table 1. Quantitative values were extracted exactly as reported by the original authors and were not recomputed. Because the studies use different models, devices, and networks, values are interpreted as indicative magnitudes for each technique rather than as head-to-head comparisons. The qualitative ratings in Table 3 were assigned by comparing the architectural properties documented in the cited works [1, 3, 17, 9].

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

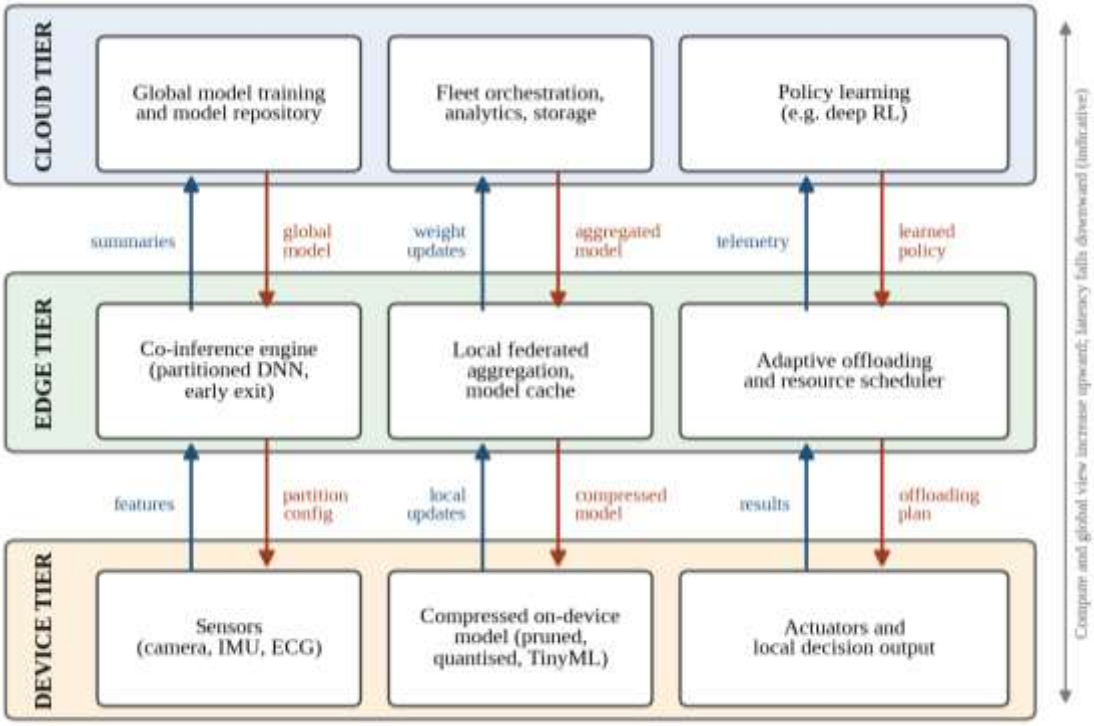


Figure 1. Reference architecture for edge–ML real-time decision systems, showing the device, edge, and cloud tiers and the upstream (blue) and downstream (red) data flows between them

Table 1. Evaluation Dimensions Used in the Analytical Framework

Dimension	Metric	Relevance to Real-Time Decisions	Typical Unit
Latency	End-to-end response time	Must not exceed the decision deadline	ms
Accuracy	Top-1 accuracy, mAP, F1	Wrong or low-confidence actions are costly	%
Energy	Energy per inference	Governs battery life of nodes	mJ
Communication	Bytes sent, training rounds	Limits cost and delay on wireless links	MB, rounds
Privacy	Raw-data exposure	Determines regulatory compliance	Qualitative
Model footprint	Parameters, memory	Must fit device memory	MB

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

RESULTS AND FINDINGS

The synthesis is presented layer by layer. Table 2 consolidates the quantitative outcomes reported for each optimisation technique, and Figure 2 compares their magnitudes on a logarithmic scale. Table 3 compares the principal deployment paradigms qualitatively, and Figure 3 illustrates how the latency budget of a decision loop shifts as computation moves towards the edge.

Model-Level Findings

Compression delivers the largest single gains in model footprint. Pruning, trained quantisation, and entropy coding together reduced AlexNet by a factor of 35 and VGG-16 by 49 [12], while efficient architectures such as MobileNets reach 70.6% ImageNet top-1 accuracy with about 4.2 million parameters and 569 million multiply-adds [15]. The two approaches are complementary, because architecture design lowers arithmetic cost and compression lowers memory traffic, which dominates energy on edge hardware [7]. Integer quantisation additionally enables execution on processors without floating-point units [13], and MCUNet shows that, with joint design, even microcontroller-class devices can sustain ImageNet-scale recognition [16]. The practical implication is that compression should be treated as a design stage rather than a post-processing step, since the memory budget of the target device determines which combination of techniques is feasible.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

Technique	Representative Study	Reported Outcome	Edge Benefit
Pruning, quantisation, coding	Han et al. [12]	AlexNet 240 MB to 6.9 MB (35×); VGG-16 49× smaller	Weights fit in on-chip memory
Efficient architecture	Howard et al. [15]	70.6% top-1; 4.2 M parameters; 569 M multiply-adds	Lower arithmetic cost
Integer-only quantisation	Jacob et al. [13]	Better accuracy–latency trade-off on mobile CPUs	No floating-point unit needed
Network and engine co-design	Lin et al. [16]	Above 70% ImageNet top-1 on a commercial microcontroller	Microcontroller-class deployment
DNN partitioning	Kang et al. [17]	≈3.1× lower latency; ≈59% less mobile energy	Adapts to bandwidth and load
Federated averaging	McMahan et al. [8]	10–100× fewer communication rounds	Raw data stay local
Adaptive federated learning	Wang et al. [21]	Aggregation frequency tuned to resource budget	Efficient use of edge resources

Table 2. Quantitative Outcomes Reported for Edge ML Optimisation Techniques

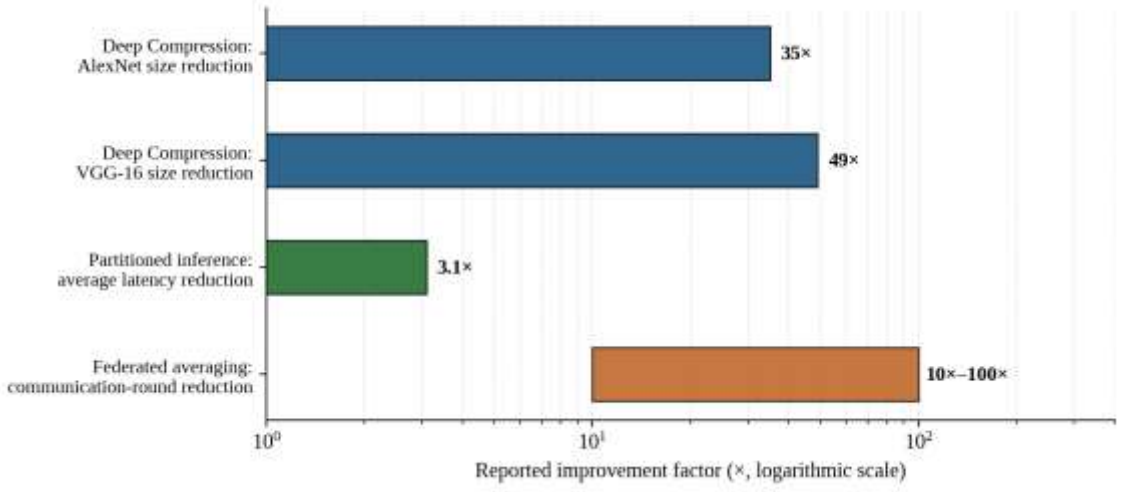


Figure 2. Improvement factors reported in the reviewed literature for model compression, partitioned inference, and federated averaging (logarithmic scale). Sources: Han et al. [12]; Kang et al. [17]; McMahan et al. [8]

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

System-Level Findings

Partitioned inference exploits the fact that intermediate activations in early convolutional layers can be larger than the input, whereas later layers produce compact representations; the best split point therefore depends on network bandwidth and server load [17]. The reported gains of about 3.1 times in latency and 59% in mobile energy relative to cloud-only execution [17] are consistent with the view that communication, not computation, is the bottleneck of cloud-centric inference. Early-exit right-sizing adds a second control knob, allowing a system to meet a hard deadline by accepting a lower-confidence prediction [18]. Taken together, the model-level and system-level results show a consistent pattern: each technique removes one bottleneck, namely memory, delay, or communication, while leaving the others in place.

Table 3. Qualitative Comparison of Deployment Paradigms for Edge ML Decision Systems

Paradigm	Latency	Bandwidth Use	Privacy	Compute Capacity
Cloud-centric ML	High	High	Low	Very high
Cloud–edge co-inference	Medium	Medium	Medium	High
Edge-only inference	Low	Low	High	Medium
On-device (TinyML)	Very low	Very low	Very high	Low
Federated edge learning	Low (inference)	Medium (training)	High	Medium

As Table 3 indicates, cloud–edge co-inference and edge-only inference occupy the middle ground: they remove most of the wide-area delay while retaining greater compute than on-device nodes. Federated edge learning differs in kind from the other paradigms because it governs where training occurs rather than where inference runs, and it therefore combines with any of the inference paradigms above it. Figure 3 illustrates this redistribution conceptually. As inference moves closer to the sensor, the network term shrinks and inference time becomes the dominant component, which is why model optimisation and architecture must be considered together [1, 3].

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

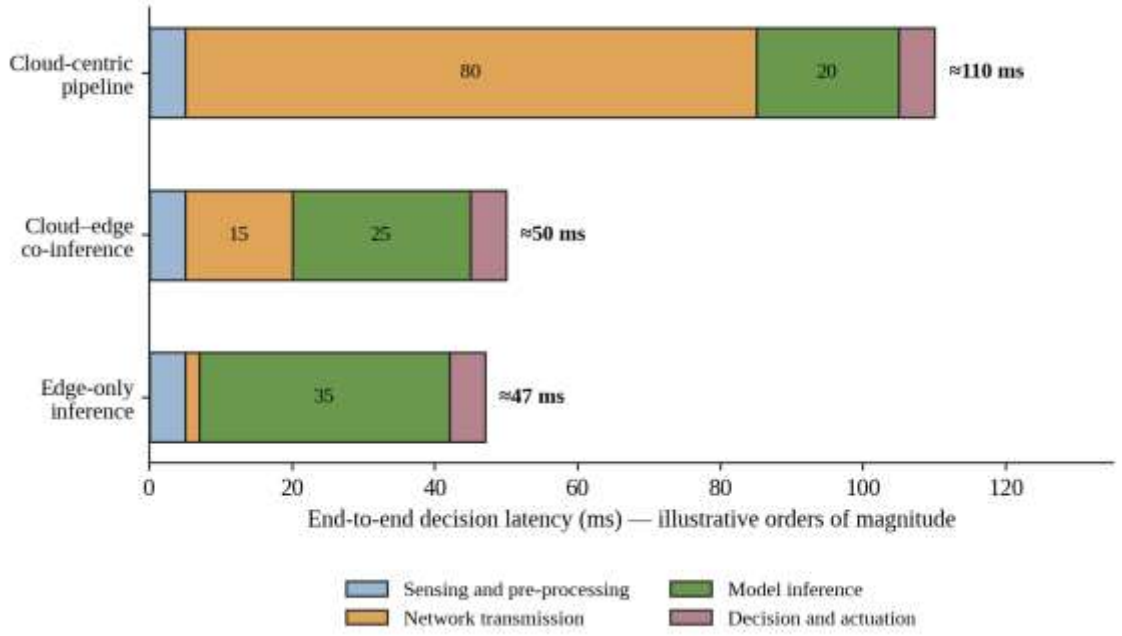


Figure 3. Conceptual decomposition of end-to-end decision latency for three deployment pipelines. Values are illustrative orders of magnitude based on the architectural arguments in [1, 3], not measured data

Learning-Level Findings

Federated learning removes the need to centralise raw data and cuts communication rounds by one to two orders of magnitude [8], yet its efficiency degrades under non-identically distributed client data and unreliable participation [9, 22]. Adaptive aggregation schedules help to match learning progress to the available computation and communication budget [21]. At the system level, production deployments require device scheduling and fault tolerance in addition to the learning algorithm itself [20].

DISCUSSION

Technical Interpretation

The findings support a decomposition of the decision loop into sensing, transmission, inference, and actuation, with the dominant term shifting as the architecture changes. In cloud-centric designs the network term dominates; at the edge it is inference time, so compression and efficient architectures become decisive [1, 12]. No technique is sufficient alone. Compression reduces memory but does not remove wireless delay, partitioning reduces delay but depends on link quality, and federated learning protects privacy but adds communication during training. Latency should also be treated as a distribution rather than a mean, because

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

real-time guarantees are violated by tail events such as link outages and server contention, which averages do not capture. Practical systems therefore combine them in hybrid device–edge–cloud designs with adaptive control, for which reinforcement learning is a promising policy engine [19, 5]. The absence of common benchmarks remains a barrier, as benchmark suites have only recently begun to cover inference and tiny devices [26, 27].

Engineering And Scientific Significance

From an engineering standpoint, the framework gives designers a structured sequence: set the decision deadline, select a deployment paradigm from Table 3, apply the optimisation techniques in Table 2 that satisfy the memory and energy limits, and then decide whether learning should be centralised or federated. The same sequence exposes trade-offs early, such as the accuracy lost to aggressive compression, which can then be weighed against the safety consequence of a missed or delayed decision. Scientifically, the convergence reframes ML as a resource-constrained systems problem in which accuracy is one metric among several, and in which algorithm, hardware, and network must be co-designed rather than optimised in isolation [7, 25].

Practical Applications

In intelligent transport, roadside edge servers can run perception and cooperative-driving models within the tight deadlines of collision avoidance, while vehicles contribute updates through federated learning [3, 24]. In industrial settings, edge inference on gateways supports predictive maintenance and closed-loop quality control without exposing production data to external networks [1]. Wearable health monitors can run compressed models on-device to raise immediate alerts and use federated learning to improve population-level models without moving patient records [9, 16]. Smart-grid operators and precision-agriculture deployments, including those in regions with intermittent connectivity such as rural India, benefit from edge nodes that keep operating when the cloud link is lost [2].

LIMITATIONS

- This is a structured narrative review rather than a formal meta-analysis; study selection reflects the judgement of the authors, and no statistical pooling of effect sizes was attempted.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

- Quantitative values are taken from the original studies, which use different models, hardware, datasets, and network conditions, so the reported factors are indicative magnitudes and not directly comparable.
- The review window of 2015 to 2021 excludes more recent developments, such as the deployment of very large transformer-based models at the edge, and Figure 3 presents illustrative values rather than measurements from a primary experiment.
- Security threats, including adversarial inputs, model poisoning, and side-channel leakage, are treated only briefly and deserve a dedicated review.

FUTURE SCOPE

- Development of standardised, open benchmarks that report latency, energy, accuracy, and communication cost together on representative edge hardware and network conditions [26, 27].
- Joint, automated optimisation of compression, partitioning, and offloading policies, for example through neural architecture search and reinforcement learning, under hard deadline constraints [19, 18].
- Robust and secure federated learning that resists poisoning and inference attacks while tolerating heterogeneous, non-identically distributed clients [22, 23].
- Energy-aware and intermittent-computing designs, supported by dedicated accelerators, to extend on-device intelligence to battery-free and energy-harvesting nodes [7, 16].
- Primary experimental validation on a physical device–edge–cloud testbed to replace the illustrative latency decomposition with measured data, including continual on-device learning under data drift.

CONCLUSION

This review examined how the convergence of edge computing and ML supports real-time intelligent decision-making. Through a four-layer framework covering architecture, model optimisation, collaborative inference, and federated learning, it showed that each layer contributes a distinct and quantifiable benefit: compression reduces model footprint by 35 to 49 times [12], partitioned inference reduces average latency by about 3.1 times [17], and federated averaging reduces communication rounds by 10 to 100 times [8]. No technique suffices alone, because the dominant source of delay moves from the network to model inference as computation approaches the sensor. Effective systems therefore combine optimised models, adaptive offloading, and privacy-preserving learning within hybrid device–

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

edge–cloud architectures. The principal obstacles are the lack of common benchmarks, the vulnerability of distributed learning to attack, and the energy limits of small devices. Addressing them through standardised evaluation, secure learning, and experimental validation will determine how reliably edge intelligence can be trusted in safety-critical, time-bound decisions.

REFERENCES

1. Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: Vision and challenges. *IEEE Internet of Things Journal.* 2016; 3(5): 637–646.
2. Satyanarayanan M. The emergence of edge computing. *Computer.* 2017; 50(1): 30–39.
3. Zhou Z, Chen X, Li E, Zeng L, Luo K, Zhang J. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE.* 2019; 107(8): 1738–1762.
4. Mao Y, You C, Zhang J, Huang K, Letaief KB. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials.* 2017; 19(4): 2322–2358.
5. Mach P, Becvar Z. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials.* 2017; 19(3): 1628–1656.
6. Deng S, Zhao H, Fang W, Yin J, Dustdar S, Zomaya AY. Edge intelligence: The confluence of edge computing and artificial intelligence. *IEEE Internet of Things Journal.* 2020; 7(8): 7457–7469.
7. Sze V, Chen YH, Yang TJ, Emer JS. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE.* 2017; 105(12): 2295–2329.
8. McMahan HB, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), PMLR.* 2017; 54: 1273–1282.
9. Li T, Sahu AK, Talwalkar A, Smith V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine.* 2020; 37(3): 50–60.
10. Wang X, Han Y, Leung VCM, Niyato D, Yan X, Chen X. Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys & Tutorials.* 2020; 22(2): 869–904.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

11. Chen J, Ran X. Deep learning with edge computing: A review. *Proceedings of the IEEE*. 2019; 107(8): 1655–1674.
12. Han S, Mao H, Dally WJ. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *International Conference on Learning Representations (ICLR).* 2016.
13. Jacob B, Kligys S, Chen B, Zhu M, Tang M, Howard A, Adam H, Kalenichenko D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2018; 2704–2713.
14. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*. 2015.
15. Howard AG, Zhu M, Chen B, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*. 2017.
16. Lin J, Chen WM, Lin Y, Cohn J, Gan C, Han S. MCUNet: Tiny deep learning on IoT devices. *Advances in Neural Information Processing Systems (NeurIPS).* 2020; 33: 11711–11722.
17. Kang Y, Hauswald J, Gao C, Rovinski A, Mudge T, Mars J, Tang L. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. *ACM SIGARCH Computer Architecture News.* 2017; 45(1): 615–629.
18. Li E, Zeng L, Zhou Z, Chen X. Edge AI: On-demand accelerating deep neural network inference via edge computing. *IEEE Transactions on Wireless Communications.* 2020; 19(1): 447–457.
19. Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. *Nature.* 2015; 518(7540): 529–533.
20. Bonawitz K, Eichner H, Grieskamp W, et al. Towards federated learning at scale: System design. *Proceedings of Machine Learning and Systems (MLSys).* 2019; 1: 374–388.
21. Wang S, Tuor T, Salonidis T, Leung KK, Makaya C, He T, Chan K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications.* 2019; 37(6): 1205–1221.
22. Kairouz P, McMahan HB, Avent B, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning.* 2021; 14(1–2): 1–210.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved

23. Lim WYB, Luong NC, Hoang DT, et al. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials.* 2020; 22(3): 2031–2063.
24. Park J, Samarakoon S, Bennis M, Debbah M. Wireless network intelligence at the edge. *Proceedings of the IEEE.* 2019; 107(11): 2204–2239.
25. Xu D, Li T, Li Y, Su X, Tarkoma S, Jiang T, Crowcroft J, Hui P. Edge intelligence: Empowering intelligence to the edge of network. *Proceedings of the IEEE.* 2021; 109(11): 1778–1837.
26. Banbury CR, Reddi VJ, Lam M, et al. Benchmarking TinyML systems: Challenges and direction. *arXiv preprint arXiv:2003.04821*. 2020.
27. Reddi VJ, Cheng C, Kanter D, et al. MLPerf inference benchmark. *Proceedings of the ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA).* 2020; 446–459.
28. Kitchenham B, Charters S. *Guidelines for performing Systematic Literature Reviews in Software Engineering (EBSE Technical Report).* Keele University and University of Durham. 2007.

***Author for Correspondence:**
Arindam Sen
E-mail: arindam.sen.cse@yahoo.com

[1]Assistant Professor, Department of Computer Science and Engineering, Kalyani Government Engineering College, West Bengal
[2]Associate Professor, Department of Computer Science and Engineering, Kalyani Government Engineering College, West Bengal

Received Date: May 11, 2025
Accepted Date: May 25, 2025
Published Date: June 09, 2025

Citation: Arindam Sen, Sneha Bhattacharya. Convergence of Edge Computing and Machine Learning for Real-Time Intelligent Decision-Making Systems: A Structured Review of Architectures, Optimisation Techniques, and Deployment Challenges. Journal of Converging Technologies in Intelligent. 2025; 1(2): 103-117p.

Page 103-117 © MANTECH PUBLICATIONS 2025. All Rights Reserved