International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 5, Sep-Oct-2026, ISSN (Online): 2395-566X

EDGE AI: Intelligent Computing At The Edge for Real-Time Applications — A Review of Architectures, Model Optimization and Deployment

Bhor Darshan Vasant[1] Prof. M. A. Auti[2] Prof. M. C. Kolse[3] Prof. S. K. Said[4]
Department Of Artificial Intelligence And Data Science Jaihind College Of Engineering Pune, Maharashtra, India;

Abstract—Edge Artificial Intelligence (Edge AI) combines machine learning with edge computing so that data are processed on or near the devices that produce them, such as sensors, cameras, smartphones, IoT nodes and embedded systems, rather than entirely in remote cloud servers. This brings decisions closer to the data source, reducing latency, bandwidth use and dependence on connectivity while improving privacy. This paper reviews the foundations and practice of Edge AI for real-time applications. It examines the edge computing paradigm and edge intelligence levels, the partitioning of neural networks between device, edge server and cloud, efficient architectures such as depthwise-separable networks, early-exit inference, and compression by pruning, quantization and knowledge distillation. Tiny machine learning on microcontrollers and standard benchmarks for measuring edge inference are discussed. Applications in smart surveillance, industrial monitoring, healthcare wearables, autonomous robots and smart cities are summarized. A layered device–edge–cloud architecture is proposed, together with research gaps, objectives, a comparison of deployment tiers and an analysis of challenges including limited resources, heterogeneity, security and model updates. The review concludes that Edge AI is essential for low-latency, private and reliable intelligent systems.

Index Terms—Edge AI, edge computing, edge intelligence, TinyML, model compression, quantization, real-time inference, Internet of Things

I. INTRODUCTION

Billions of devices now capture images, sound and sensor readings continuously. Sending all of this data to the cloud is costly in bandwidth, adds delay and exposes personal information. For many applications, such as detecting an intruder on a camera or stopping a machine before it fails, a decision delayed by even a second may be too late.

Satyanarayanan [1] described the emergence of edge computing, in which computing resources are placed close to users and devices, and Shi *et al.* [2] outlined its vision and challenges. Combining this paradigm with machine learning gives Edge AI, which Zhou *et al.* [3] call "the last mile" of artificial intelligence.

This paper reviews Edge AI architectures, model optimization techniques and applications, and proposes a reference architecture for real-time systems.

II. RELATED WORK

Kang *et al.* [4] proposed Neurosurgeon, which automatically splits a deep network between a mobile device and the cloud at the layer that minimizes latency or energy, showing that partitioning can outperform both all-device and all-cloud execution. Teerapittayanon *et al.* [5]

introduced BranchyNet, in which easy inputs exit early from intermediate classifiers, reducing average inference time.Efficient architectures are central. MobileNets [6] and MobileNetV2 [7] use depthwise-separable convolutions and inverted residual blocks to cut computation for mobile vision.

Han *et al.* [8] showed that pruning, quantization and coding can shrink networks dramatically, and Hinton *et al.* [9] introduced knowledge distillation to train compact students.At the smallest scale, Lin *et al.* [10] co-designed network architecture and inference engine in MCUNet to run ImageNet-scale models on microcontrollers with only hundreds of kilobytes of memory.

Banbury *et al.* [11] defined MLPerf Tiny, a benchmark suite for measuring accuracy, latency and energy of ultra-low-power inference.

Chen and Ran [12] and Li *et al.* [13] reviewed deep learning at the edge and highlighted the need to balance accuracy, latency and energy across heterogeneous hardware. Table I summarizes these works.

© 2026 IJSRET

TABLE I Summary of Literature Survey

Ref.	Author(s) / Year	Technique / Approach	Key Contribution	Limitation / Gap
[1]	Satyanarayanan, 2017	Edge computing paradigm	Motivated computing near users	Conceptual
[3]	Zhou et al., 2019	Edge intelligence survey	Defined levels of edge intelligence	Survey only
[4]	Kang et al., 2017	Neurosurgeon DNN partitioning	Lower latency and energy via splitting	Depends on network conditions
[5]	Teerapittayanon et al., 2016	BranchyNet early exits	Faster average inference	Extra training complexity
[7]	Sandler et al., 2018	MobileNetV2 inverted residuals	Efficient mobile vision backbone	Lower accuracy than large CNNs
[8]	Han et al., 2016	Deep compression	Large size reduction	Sparse hardware support needed
[10]	Lin et al., 2020	MCUNet co-design	ImageNet-scale models on MCUs	Specialized toolchain
[11]	Banbury et al., 2021	MLPerf Tiny benchmark	Standard tiny ML measurement	Limited task set

III. PRELIMINARIES

3.1 Device, Edge And Cloud

Computation can occur on the device itself, on a nearby edge server such as a gateway or base station, or in the cloud. Fig. 1 summarizes the tiers and the techniques used at each.

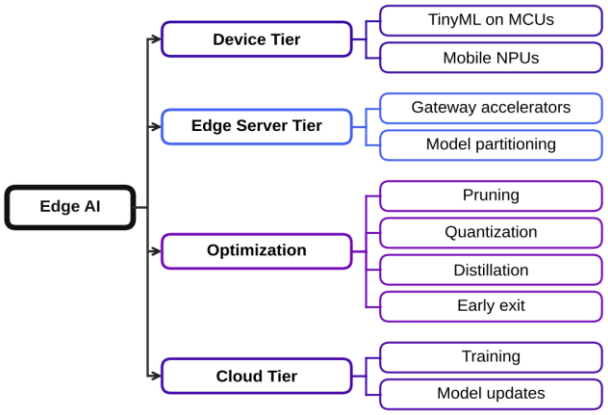


Fig. 1. Edge AI deployment tiers and enabling techniques.

3.2 Levels Of Edge Intelligence

Zhou *et al.* [3] define levels ranging from cloud training with edge inference, through cooperative training and inference, to fully on-device training and inference. Most current applications train in the cloud and infer at the edge.

IV. PROBLEM STATEMENT AND OBJECTIVES

4.1 Identified Research Gaps

- Heterogeneous edge hardware makes it hard to deploy one model everywhere.
- Updating models on thousands of devices securely and reliably is not well solved.
- On-device training and personalization remain limited by memory and energy.
- Benchmarks often report accuracy without real-world latency and power.
- Security of edge devices and models against tampering is under-addressed.

4.2 Objectives Of The Review

- To explain the edge computing paradigm and levels of edge intelligence.
- To review model partitioning, efficient architectures and compression techniques.
- To propose a device–edge–cloud architecture for real-time AI applications.
- To compare deployment tiers in latency, privacy and capability.
- To identify challenges and future research directions.

© 2026 IJSRET

4.3 Proposed Framework

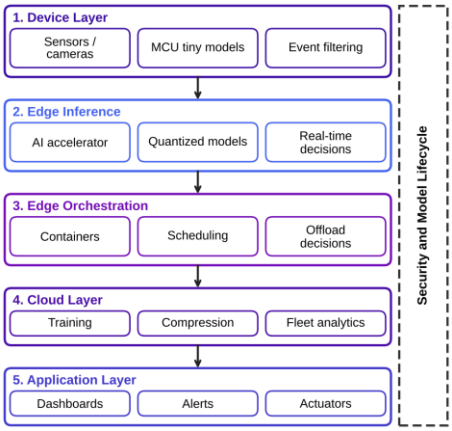


Fig. 2 shows the proposed layered architecture.

Fig. 2. Proposed device–edge–cloud architecture for real-time Edge AI.

4.4 Device Layer

Sensors and cameras capture data; microcontrollers run tiny models for wake-up and filtering, sending only relevant events upward.

4.5 Edge Inference Layer

A gateway with an AI accelerator runs quantized models for detection, classification and anomaly analysis in real time.

4.6 Edge Orchestration Layer

Containers manage models, schedule workloads, collect metrics and decide when to offload to the cloud.

4.7 Cloud Layer

The cloud trains and compresses models, aggregates fleet statistics and distributes updates.

4.8 Application Layer

Dashboards, alerts and actuators act on decisions, for example locking a door or stopping a conveyor.

V. WORKING METHODOLOGY

Fig. 3 shows the deployment workflow. Total response time for an edge application is

$$T = T_sense + T_pre + T_infer + T_act \quad (1)$$

and offloading to the cloud is chosen only when T_upload + T_cloud + T_download is smaller than local inference time while meeting privacy rules. Quantization maps a real value r to an 8-bit integer q using a scale s and zero-point z:

$$q = round(r / s) + z \quad (2)$$

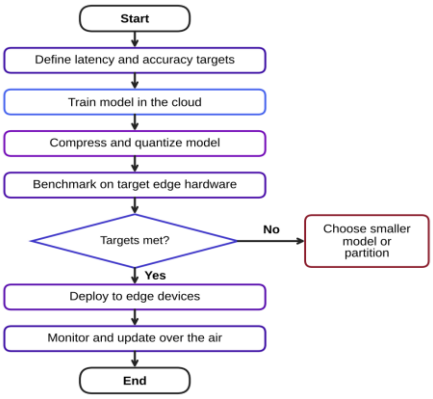


Fig. 3. Workflow for deploying an Edge AI application.

VI. DISCUSSION

6.1 Comparison Of Approaches

Table II compares deployment tiers. Devices offer the lowest latency and best privacy but limited capability; edge servers balance capability and latency; the cloud provides the most power but the highest delay.

TABLE II Comparison of Deployment Tiers

Tier	Latency	Privacy	Capability
Microcontroller	Very low	Very high	Tiny models only
Smartphone / NPU	Low	High	Medium models
Edge server	Low–Medium	Medium–High	Large models
Cloud	High	Lower	Very large models

6.2 Open Challenges And Future Work

Table III lists challenges. Future directions include federated learning across edge devices, neuromorphic and in-memory computing hardware, small language models running locally, and standard frameworks for secure over-the-air model updates.

TABLE III Challenges and Mitigation

Challenge	Mitigation
Limited memory and power	Compression; efficient architectures
Hardware heterogeneity	Standard runtimes (ONNX, TFLite)
Model updates	Secure over-the-air updates
Device security	Secure boot; encrypted models
On-device learning	Federated and incremental learning

© 2026 IJSRET

VII. CONCLUSION AND FUTURE WORK

Edge AI brings intelligence to where data is produced, enabling real-time, private and bandwidth-efficient applications. The reviewed literature shows that network partitioning, efficient architectures, compression and tiny ML make deep learning practical on constrained hardware. Managing heterogeneity, updates and security will determine how widely Edge AI can be deployed.

7.1 Acknowledgment

The authors express their sincere gratitude to the Department of Artificial Intelligence and Data Science, Jaihind College of Engineering, Pune, for providing the facilities, guidance and encouragement needed to carry out this work. The authors also thank the faculty members and colleagues whose valuable suggestions helped improve the quality of this review.

REFERENCES

1. M. Satyanarayanan, "The emergence of edge computing," Computer, vol. 50, no. 1, pp. 30–39, 2017.
2. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet Things J., vol. 3, no. 5, pp. 637–646, 2016.
3. Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," Proc. IEEE, vol. 107, no. 8, pp. 1738–1762, 2019.
4. Y. Kang et al., "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," in Proc. 22nd Int. Conf. Archit. Support Program. Lang. Oper. Syst. (ASPLOS), 2017, pp. 615–629.
5. S. Teerapittayanon, B. McDanel, and H. T. Kung, "BranchyNet: Fast inference via early exiting from deep neural networks," in Proc. 23rd Int. Conf. Pattern Recognit. (ICPR), 2016, pp. 2464–2469.
6. A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, 2017.
7. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF CVPR, 2018, pp. 4510–4520.
8. S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in Proc. ICLR, 2016.
9. G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv:1503.02531, 2015.
10. J. Lin, W.-M. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, "MCUNet: Tiny deep learning on IoT devices," in Proc. NeurIPS, vol. 33, 2020, pp. 11711–11722.
11. C. Banbury et al., "MLPerf Tiny benchmark," in Proc. NeurIPS Datasets and Benchmarks Track, 2021.
12. J. Chen and X. Ran, "Deep learning with edge computing: A review," Proc. IEEE, vol. 107, no. 8, pp. 1655–1674, 2019.
13. H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," IEEE Netw., vol. 32, no. 1, pp. 96–101, 2018.

© 2026 IJSRET