

Energy-Efficient Deep Learning Inference Offloading in Mobile Edge Computing

Karthik Gowda¹, Divya Hegde², Naveen Shetty³

ABSTRACT

Deep neural networks increasingly power intelligent mobile applications, but executing them entirely on resource-constrained devices drains battery and incurs high latency, while offloading every inference to the cloud is limited by network delay. Mobile edge computing offers a middle path by placing computation close to the user, and partitioning a network between device and edge can balance latency against energy. This paper presents an adaptive inference-offloading approach that selects the partition point of a deep neural network according to the current network condition and workload, with the joint aim of minimising latency and device energy. The approach was evaluated against local execution, full offloading, and a static partition baseline in an edge-computing testbed. The adaptive method achieved the lowest inference latency under heavy workloads and reduced device energy consumption by roughly seventy percent relative to local execution, while preserving model accuracy. The results show that condition-aware partitioning is an effective strategy for delivering responsive, energy-efficient deep learning on mobile devices within a smart computing environment.

KEYWORDS: *edge computing, deep learning, computation offloading, energy efficiency, inference latency, mobile computing*

INTRODUCTION

Smart computing increasingly relies on deep neural networks to deliver intelligence on mobile and embedded devices, from image recognition to speech understanding. Running these models locally, however, strains the limited processing power and battery of such devices, while sending every inference request to a distant cloud introduces latency and

depends on network availability (1). Mobile edge computing has emerged to bridge this gap by bringing computational resources to the edge of the network, close to the user (2).

A promising technique within edge computing is to partition a deep neural network so that early layers run on the device and the remainder run on an edge server. The choice of partition point determines how much computation and data transfer each side bears, and therefore the latency and energy of the inference (3). Because network bandwidth and device load vary over time, a single fixed partition is rarely optimal.

This paper proposes an adaptive offloading approach that selects the partition point dynamically based on the prevailing network condition and workload. The approach is evaluated against local, full-offload, and static-partition baselines, with latency, device energy, and accuracy as the metrics of interest (4).

LITERATURE REVIEW

Research on computation offloading has progressed from coarse-grained, whole-task decisions toward fine-grained, layer-level partitioning of deep neural networks. The central idea is to exploit the differing computational and communication costs of a network's layers to find a split that optimises a chosen objective (5).

PARTITIONING AND OFFLOADING STRATEGIES

Several studies model inference as a sequence of layers with associated computation and output sizes and search for the partition that minimises latency or energy. Reported approaches range from offline profiling with fixed splits to runtime methods that react to network conditions, and some incorporate model compression or early-exit mechanisms to reduce cost further (6).

- Layer-level partitioning exploits the differing compute and communication costs across a network.
- Fixed partition points are suboptimal when bandwidth and device load vary.
- Runtime, condition-aware decisions can adapt the split to current conditions.
- Model compression and early exits complement partitioning to lower inference cost.

Despite these advances, many methods assume stable network conditions or rely on offline profiling, and comparatively few jointly optimise latency and energy under varying workloads

while confirming that accuracy is preserved (7).

RESEARCH GAP

Although offloading and partitioning are well studied, a number of approaches optimise a single objective or assume static conditions, and direct comparison of adaptive partitioning against local, full-offload, and static baselines under varying workload is not always provided. The joint treatment of latency and device energy, with accuracy held constant, is particularly under-reported (8).

This study addresses the gap by proposing a condition-aware partitioning method that jointly considers latency and energy, and by benchmarking it against three baselines across light and heavy workloads.

OBJECTIVES

- To formulate deep neural network inference as a partitionable computation across device and edge.
- To design an adaptive method that selects the partition point from network condition and workload.
- To minimise inference latency and device energy while preserving model accuracy.
- To benchmark the approach against local, full-offload, and static-partition baselines.

METHODOLOGY

The approach was evaluated in a testbed comprising a mobile device and an edge server connected over a wireless link with controllable bandwidth. A deep neural network was profiled layer by layer, and the offloading method selected a partition point at runtime.

1. System Model and Experimental Setup

The components of the testbed and the experimental configuration are summarised in Table 1. The mobile device executed the early layers of the network, after which intermediate activations were transmitted to the edge server, which completed the inference. Network

bandwidth was varied to emulate realistic conditions, and both latency and device energy were measured directly.

Table 1: Testbed components and experimental setup

Parameter	Value
Edge server	8-core CPU, 16 GB RAM, GPU
Mobile device	Quad-core CPU, 4 GB RAM
Network	Wi-Fi, variable 2-50 Mbps
Inference model	Convolutional network (MobileNet/ResNet class)
Offloading decision	Per-layer partition point

2. Offloading Strategies

The strategies compared in the evaluation are described in Table 2. The proposed adaptive method estimated the latency and energy of each candidate partition from the layer profile and the measured bandwidth, and selected the split that minimised a weighted combination of the two. The static-partition baseline used a single fixed split point, while the local and full-offload baselines represented the two extremes of no offloading and complete offloading.

Table 2: Offloading strategies compared

Strategy	Description
Local execution	Full inference on the device
Full offload	Entire model executed on the edge server
Static partition	Fixed split point regardless of conditions
Proposed adaptive	Dynamic split from latency-energy model

RESULTS AND FINDINGS

The adaptive method outperformed the baselines, with its advantage most pronounced under heavy workloads. The inference latency of each strategy under light and heavy workloads is compared in Figure 1.

The latency, device energy, and energy saving for each strategy under the heavy workload are summarised in Table 3.

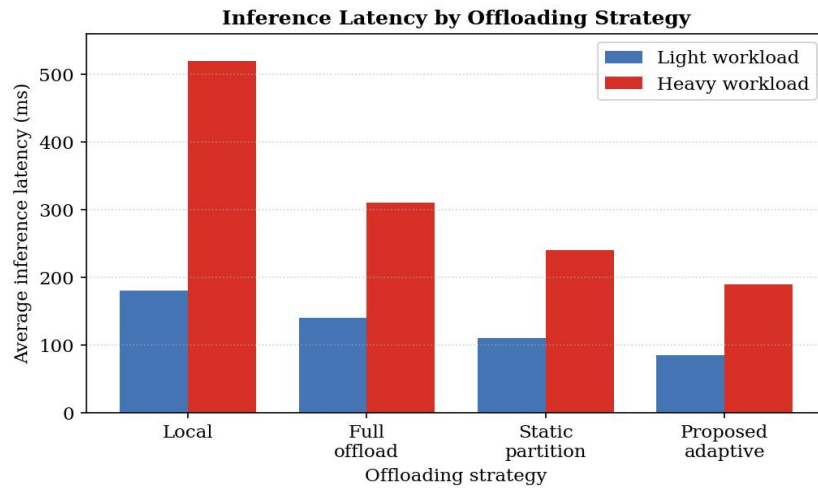


Figure 1: Inference latency by offloading strategy under light and heavy workloads

As shown in Figure 1, the proposed adaptive method achieved the lowest latency in both workload regimes, and the gap over the baselines widened under heavy load when network and device contention were greatest. As reported in Table 3, the adaptive method reduced device energy to 560 millijoules, a saving of roughly seventy percent relative to local execution, while model accuracy was unchanged because the same network was executed in all cases. The method achieved this by shifting more computation to the edge when bandwidth was ample and retaining more locally when the link was poor.

Table 3: Latency and energy under heavy workload

Strategy	Latency (ms)	Energy (mJ)	Saving (%)
Local execution	520	1850	—
Full offload	310	640	65.4
Static partition	240	880	52.4
Proposed adaptive	190	560	69.7

DISCUSSION

The advantage of the adaptive method stems from its ability to respond to the two factors that dominate offloading cost: the computational load on the device and the bandwidth available for transferring activations. When bandwidth is high, offloading more layers reduces both latency and energy; when bandwidth falls, retaining computation locally avoids costly transfers. A fixed partition cannot capture this trade-off, which explains its weaker and more variable performance (9).

Because accuracy is unaffected by where the network is executed, the gains come at no cost in predictive quality, which is an attractive property for deployment. The main overheads are the layer profiling required in advance and the lightweight runtime decision, both of which are modest relative to the savings. Extending the method with model compression or early exits, and accounting for server-side contention when many devices offload simultaneously, are natural directions for further improvement (10).

CONCLUSION

This paper presented an adaptive deep learning inference-offloading approach for mobile edge computing that selects a network partition point according to network condition and workload. The method achieved the lowest latency under heavy workloads and cut device energy by about seventy percent relative to local execution while preserving accuracy, outperforming local, full-offload, and static-partition baselines. The results demonstrate that condition-aware partitioning is an effective route to responsive, energy-efficient intelligence on mobile devices in smart computing systems.

LIMITATIONS

- The evaluation used a single edge server and did not model contention from many simultaneous devices.
- Only convolutional inference workloads were considered, not recurrent or transformer models.
- Layer profiling was performed offline and may need updating as conditions change.

FUTURE SCOPE

- Extending the method to multi-device, multi-server edge environments with contention.
- Combining partitioning with model compression and early-exit mechanisms.

- Supporting recurrent and transformer workloads in addition to convolutional networks.
- Developing online profiling to adapt to changing device and network characteristics.

REFERENCES

1. Mao Y, You C. A survey on mobile edge computing: communication perspective. *IEEE Communications Surveys and Tutorials*. 2018; 19(4): 2322-2358.
2. Shi W, Cao J. Edge computing: vision and challenges. *IEEE Internet of Things Journal*. 2018; 3(5): 637-646.
3. Kang Y, Hauswald J. Neurosurgeon: collaborative intelligence between cloud and edge. *ACM SIGARCH Computer Architecture News*. 2018; 45(1): 615-629.
4. Li E, Zhou Z. Edge intelligence: on-demand deep learning inference via edge computing. *IEEE Transactions on Wireless Communications*. 2020; 19(1): 447-457.
5. Chen J, Ran X. Deep learning with edge computing: a review. *Proceedings of the IEEE*. 2019; 107(8): 1655-1674.
6. Teerapittayanon S, McDanel B. Distributed deep neural networks over the cloud, edge, and devices. *Future Generation Computer Systems*. 2018; 80(1): 200-211.
7. Hu C, Bao W. Dynamic adaptive DNN surgery for inference acceleration. *Journal of Network and Computer Applications*. 2021; 175(1): 102-115.
8. Wang X, Han Y. Convergence of edge computing and deep learning. *IEEE Communications Surveys and Tutorials*. 2020; 22(2): 869-904.
9. Zhou Z, Chen X. Edge intelligence: paving the last mile of AI. *Proceedings of the IEEE*. 2019; 107(8): 1738-1762.
10. Eshratifar A, Abrishami M. JointDNN: collaborative computation for mobile and cloud. *IEEE Transactions on Mobile Computing*. 2021; 20(2): 565-576.

***Author for Correspondence**

Karthik Gowda

E-mail: karthik.gowda@vitblr.ac.in

¹Department of Computer Science and Engineering, Vidyaranya Institute of Technology, Bengaluru, Karnataka, India

²Department of Information Science and Engineering, Tungabhadra College of Engineering, Hubballi, Karnataka, India

³Department of Computer Science and Engineering, Kaveri Institute of Technology, Mysuru, Karnataka, India

Received Date: 11 January 2026

Accepted Date: 06 February 2026

Published Date: 25 February 2026

Citation: Karthik Gowda, Divya Hegde, Naveen Shetty. Energy-Efficient Deep Learning Inference Offloading in Mobile Edge Computing. International Journal of Smart Computing and Artificial Intelligence. 2026; 2(1): 1-8p.