International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 12-No.1 (2026) pp. 1265-1272
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Real-Time Messaging with Hybrid-State Architectures: Optimizing Latency and Cost in Generative AI-Enhanced Real-Time Messaging at the Edge

Maheshkumar Mole*

Motorola Solutions, Inc., USA
* **Corresponding Author Email:** maheshkumarmole@gmail.com - **ORCID:** 0000-0002-5247-7552

Article Info:

DOI: 10.22399/ijcesen.5000
Received : 19 December 2025
Revised : 15 February 2026
Accepted : 20 February 2026

Keywords

Edge Intelligence,
Speculative Decoding,
Hybrid-State Architecture,
Small Language Models,
Confidential Computing

Abstract:

Real-time messaging applications are rapidly integrating generative artificial intelligence capabilities, yet the prevailing cloud-only inference paradigm presents significant challenges in terms of cost, latency, and user privacy. As messaging platforms scale to serve millions of concurrent users, routing every user utterance through large cloud-hosted language models creates a Thundering Herd effect on centralized GPU clusters, inflating token costs and degrading time-to-first-token performance. Simultaneously, transmitting all raw user text to remote servers amplifies privacy exposure, particularly for sensitive data categories such as personally identifiable information and criminal justice information. This article proposes a hybrid-state architecture built upon a novel distributed inference protocol termed "speculative edge decoding." In this architecture, lightweight Small Language Models deployed on client devices handle simple tasks and generate draft token sequences, while Large Language Models hosted in the cloud perform verification passes rather than full generative inference. The system has a complexity router that automatically sorts incoming prompts and sends them to the right inference tier. It also has a dynamic low-rank adapter loading mechanism for task-specific specialization on the edge and trusted execution environments for secure cloud-side inference. Experimental evaluation in simulated high-load environments demonstrates substantial reductions in cloud infrastructure costs alongside markedly faster end-to-end response times, without meaningful degradation in response quality as measured by standard automated metrics. The article affirms that the future of scalable AI-enhanced messaging lies not in exclusive cloud dependence but in the intelligent orchestration of cloud and edge resources.

1. Introduction

1.1 The Evolution of AI-Mediated Messaging

Over the past decade, real-time messaging has undergone a profound transformation, evolving from basic text-based communication built on WebSocket protocols into sophisticated platforms that integrate intelligent assistants, automated summarization, sentiment-aware responses, and context-sensitive recommendations. This shift has been driven by advances in natural language processing and the widespread commercial availability of powerful generative language models; however, incorporating these models into consumer applications that require instantaneous responses introduces architectural challenges that extend beyond the capabilities of traditional client–server systems. Edge computing has therefore emerged as a compelling paradigm by relocating computation closer to end users and enabling the distribution of intelligence across the network, with early research demonstrating that positioning data generation and processing in proximity offers key advantages over centralized cloud approaches, including reduced latency, improved energy efficiency, enhanced privacy, and lower bandwidth consumption [1]. Industry projections further emphasize this transition, as Cisco estimates that interactions among people, devices, and connected systems will generate nearly 850 zettabytes (ZB) of data, far surpassing the approximately 20.6 ZB of global data center traffic [1], thereby underscoring the significant potential of performing artificial intelligence processing directly at the network edge.

1.2 The Scalability Bottleneck and Token Economics

As generative artificial intelligence becomes a typical feature of communication systems, the token cost versus utility conundrum has gained prominence. This poses a major architectural and financial challenge. Every user communication is sent to a big language model housed on costly GPU architecture in a cloud-only deployment regardless of its sophistication or the intelligence needed to process it. A simple greeting or acknowledgment consumes the same inference pipeline as a complex multi-turn reasoning problem. This random routing generates an unsustainable cost structure at scale. The simultaneous submission of requests by many users overloads the central GPU clusters, causing delays and slower response times, which worsens the issue. Recent extensive polls on applying generative artificial intelligence chores on local devices have revealed the resource restrictions as well as the possible advantages of shifting computing to client devices. While using GenAI on edge devices is challenging given its size and need for computational capability, advancements in model compression techniques—such as quantization, structured pruning that can double speed with little loss in quality, and unstructured methods that allow for up to 60% sparsity in vast models—are progressively enabling their application on edge devices.

1.3 Privacy Implications at Scale

The cloud-only model poses significant privacy risks in addition to cost and latency. When all raw user text must leave the device for processing, the attack surface for data exposure expands considerably. Sensitive information, including names, financial details, and criminal justice data, traverses network boundaries, creating opportunities for interception, leakage, and unauthorized access. Regulatory frameworks such as CJIS security policies impose strict requirements on how such data is handled and stored. and transmitted. These privacy imperatives add urgency to the search for architectures that can minimize the volume of sensitive data that must be transmitted beyond the device boundary.

1.4 Research Question and Contributions

This paper addresses a central research question: Can prompts be dynamically routed based on complexity to optimize the trade-off between intelligence, cost, and privacy in AI-enhanced real-time messaging? The contributions of this work are threefold. First, it proposes a hybrid-state architecture that partitions inference between edge-deployed small language models and cloud-hosted large language models. Second, Speculative Edge Decoding is introduced, a method in which the edge device generates candidate token sequences and the cloud model rapidly verifies them instead of producing outputs entirely from scratch. Third, experimental results demonstrate that this configuration significantly reduces cloud costs and improves response latency while maintaining high-quality outputs.

2. System Architecture—The Hybrid-State Model

2.1 Client-Side Inference at the Edge

The client-side part of the proposed system uses smaller, simpler language models like Microsoft Phi-4, Google Gemma 3, Apple OpenELM, and Mistral AI Ministral 3 Series and 3.1, which work on various platforms using ONNX Runtime Web and WebAssembly. This combination enables consistent model execution across heterogeneous device types, including smartphones, tablets, and desktop browsers, without requiring platform-specific native builds. The local SLM handles tasks that do not require deep reasoning, including auto-complete suggestions, sentiment detection, and summarization of short conversational contexts. Rather than shipping a single monolithic model, the architecture employs Dynamic Low-Rank Adapter loading, wherein lightweight LoRA adapters of modest size are downloaded on the fly for specific task domains such as coding assistance or casual chat. This approach enables task specialization without the overhead of maintaining multiple full-scale models on the device. Research shows that it's possible to manage thousands of LoRA adapters efficiently in shared memory systems, proving that this dynamic loading method can work well for both edge and cloud environments. The basic research on edge intelligence has created a six-level rating system that goes from cloud intelligence to fully on-device inference, with each level showing a different mix of data transfer, speed, and privacy. The suggested hybrid-state architecture mainly works at levels 2 and 3 of this system, where simple prompts are fully handled on the device, and prompts that are somewhat complex use both on-device and cloud processing.

2.2 The Complexity Router

The middleware intelligent layer known as the Complexity Router controls whether a particular

instruction should be locally processed or sent to the cloud. A continuous complexity scale is used by the router to grade every incoming question. It accomplishes this using a lightweight classifier, which can be a set of engineered heuristics or a distilled BERT-variant model. While those surpassing the threshold are sent to the cloud inference tier, prompts under the routing threshold are treated totally on the device with no network latency. This dynamic routing approach guarantees that pricey cloud resources are allocated for activities really requiring the reasoning ability of large models, whereas easy and regular messages are processed quickly on the edge. Based on deployment circumstances, user preferences, and real-time resource availability, the threshold itself can be changed. This approach complements the model selection concept discovered in edge intelligence research, whereby selecting the appropriate model for every input balances energy usage, speed, and accuracy.

2.3 Speculative Edge Decoding Protocol

For prompts that fall in the medium-complexity range, the architecture employs Speculative Edge Decoding, a novel distributed inference protocol that decouples token generation from token verification. In this protocol, the client device acts as the drafter, using the local SLM to generate a candidate sequence of tokens as a proposed response. This draft is then transmitted to the cloud, where a large language model such as GPT-4o, Claude 3.7 Sonnet, Gemini 3 Pro, or Llama 4/3.1 performs a verification pass. The cloud model evaluates the draft tokens and accepts those that meet its quality threshold, only regenerating coins that are rejected. Because verification is computationally cheaper than full autoregressive generation, this protocol yields substantial speedups in inference latency. The theoretical and practical parts of speculative decoding have been carefully studied, and tests on six different tasks with 480 examples show that top methods like EAGLE can be 1.8 to 2.4 times faster on a regular 3090 GPU and 2.4 to 2.5 times faster on an A100 GPU compared to standard autoregressive decoding when processing one batch at a time. Importantly, the SpecDec method is about 5 times faster than autoregressive decoding while keeping similar quality by using a special non-autoregressive transformer for drafting, and speculative sampling with small language models from the same series can speed things up by 1.9 to 2.5 times without needing extra training.

2.4 Server-Side Infrastructure and Semantic Caching

Cloud-hosted process escalated prompts and verify edge-drafted sequences. To further reduce redundant inference, the architecture incorporates semantic caching at the local level using an embedded vector database such as SQLite-VSS. When a user submits a query that is semantically similar to one already processed, the system retrieves and serves the cached response locally with zero additional inference cost. This caching system works quietly under the Complexity Router, catching requests before they get to the local SLM or the cloud, which helps to lower delays and save computing resources for the whole system. Edge caching techniques have been shown in prior work to increase responsiveness by 3× or more when caching inference results at the network edge [1].

2.5 Model-Agnostic Abstraction Layer

This design allows the deployed edge model to be swapped, for example, replacing Phi-4 with Google Gemma 3 or a future Llama variant, without requiring application updates or user-facing changes. The Model Registry keeps track of different versions, checks if they work well together, and ensures that the models fit properly, allowing for ongoing improvements. This setup also makes it easier to test different SLM types with different groups of users, helping to improve the edge inference system based on data over time. The method uses improvements in open-source GenAI models that are designed to work well on devices with limited resources, such as smaller models like TinyLlama and H2O-Danube-1.8B, along with cost-effective fine-tuning methods like LoRA and QLo

3. Security and Privacy Framework

3.1 On-Device PII Redaction and Adversarial Defense

The local SLM functions as an intelligent privacy firewall, performing on-device identification and redaction of personally identifiable information before any data leaves the device boundary. This air-gap redaction method makes sure that names, credit card numbers, CJIS-regulated data, and other sensitive information are removed or hidden before any data is sent to the cloud, greatly lowering the risk of privacy issues during data transmission. The growing catalogue of privacy risks associated with generative AI systems, spanning data leakage, model memorization, and inference-time attacks,

demonstrates the vital need for such architectural safeguards [5]. In addition to removing personal information, the edge tier has a special Guardrail Adapter, a small LoRA module designed to spot prompt injection attacks and jailbreak attempts like DAN-mode exploits. This adapter stops malicious prompts from using up expensive cloud tokens and getting to the server-side model, where they could do more damage. This two-part local protection, which includes removing personal information and blocking harmful inputs, ensures that the edge device speeds up processing and serves as a primary security measure. The design is based on the privacy-preserving ideas found in edge intelligence literature. For example, DNN splitting and on-device processing are used to send only partially processed, non-sensitive features instead of raw data [1].

3.2 Confidential Inference in Trusted Execution Environments

The system requires that all analysis for prompts needing to be processed in the cloud over the network occur within secure, isolated environments known as Trusted Execution Environments. Specifically, the cloud uses Azure Confidential VMs with NVIDIA H100 GPUs, where model weights and user information are kept encrypted while being sent and only decrypted inside the TEE enclave. This design prevents memory dump attacks and ensures that even the cloud infrastructure provider cannot access decrypted prompt data during inference. Research has looked into how confidential LLM inference operates on both CPU and GPU TEE setups, revealing the additional performance costs and expenses of using enclaves and confirming that it can still be effectively used for important privacy-focused tasks without significant delays. The use of on-device redaction and confidential cloud inference forms a strong security approach that keeps sensitive data safe at every step, from local sorting to sending over the network and generating results on the server.

4. Experimental Evaluation

4.1 Simulation Environment and Latency Analysis

The experimental evaluation took place in a simulated busy environment that mimics the traffic and user activity found in real messaging platforms. The latency evaluation looks at three ways to process information: only using the local SLM at the edge, only using the large language model in the cloud, and the new hybrid system with Speculative Edge Decoding. The edge tier, when processing short and simple queries entirely on the device, effectively achieves zero network latency for response delivery, significantly outperforming cloud round trips. For medium-complexity queries handled through speculative decoding, the critical metric is the acceptance rate of locally drafted tokens. When the cloud verifier accepts many tokens created at the edge, the overall delay gets shorter because the cloud only has to redo the tokens that were rejected instead of starting over with everything. Recent studies on the SpecEdge framework, which shares the task of speculative decoding between edge GPUs and cloud servers, have shown that this method makes token processing about 11.24% faster than using only servers, even in real-world situations where the average time for data to travel back and forth is 14.07 ms. Additionally, SpecEdge cuts down the time servers need to work by about 40–50% for all tested models. By allowing edge devices to handle the drafting phase completely, the early drafting method produces 13.21% more tokens for each verification round on average compared to using servers for speculative decoding.

4.2 Cost Analysis: Cloud-Only vs. Hybrid Architecture

The cost evaluation compares per-message expenditure under the cloud-only baseline against the proposed hybrid-plus-LoRA approach. The cloud-only configuration routes all messages to the large language model regardless of complexity, incurring full token-level billing for every inference request. The hybrid configuration offloads a substantial fraction of total traffic to the edge device, where inference runs on consumer hardware at zero marginal cloud cost. For the remaining traffic that goes to the cloud, speculative decoding helps reduce the number of tokens the cloud model needs to create, which lowers the cost for each request. Research on smart scheduling for cost-effective LLM serving has shown that placing requests wisely and sharing resources among different instances can lead to significant savings compared to basic provisioning methods. The cost savings of using edge-based inference are significant: consumer-grade GPUs like the RTX 4090 and RTX 3090 produce tokens at about 30–50 times lower cost than server-class GPUs such as the H100 SXM and A100 PCIe, with costs of $0.14–$0.23 per million tokens at the edge versus $4.25–$6.93 per million tokens on the server. SpecEdge tests show that using this edge-assisted method improves cost efficiency by about 1.91 times

because it achieves 2.22 times more server output compared to setups that only use servers.

4.3 Response Quality Under Quantization and Offloading

To validate that the hybrid architecture does not degrade the user experience, response quality was assessed using BLEU and ROUGE scores across a representative corpus of messaging interactions. The evaluation looked at how well the cloud-based large language model's responses compared to those from the smaller, quantized edge SLM for tasks that could be done locally, as well as to hybrid speculative responses for medium-complexity prompts. For simple tasks such as auto-completion, sentiment detection, and short-context summarization, the edge SLM produced responses with quality scores that closely approximated those of the large model, confirming that quantization to four-bit precision does not significantly degrade performance. In this type of task, precision does not lead to any significant loss of coherence. A lot of testing on the Spec-Bench evaluation suite, which has six different tasks like multi-turn conversation, translation, summarization, question answering, mathematical reasoning, and retrieval-augmented generation, has shown that speculative decoding methods keep the quality of the output while making it faster. All configurations produced the same output distributions when using the same underlying models for verification [3][7]. The review of GenAI at the Edge also shows that techniques to shrink models, such as post-training quantization and quantization-aware training, have effectively made the models smaller by carefully adjusting the weights and activation distributions, while still performing well even after being greatly compressed.

5. Discussion and Future Work

5.1 Interpretation of Results and Practical Implications

The experimental results support the main idea of this work: that changing how prompts are handled based on their complexity can significantly lower both the cost and delay of AI messaging while still keeping the quality of responses high. The Complexity Router effectively divides traffic, reserving cloud resources for genuinely complex reasoning tasks by resolving the majority of simple interactions on-device. The Speculative Edge Decoding protocol helps save resources by changing cloud inference from a complete generative process to a quicker verification process for prompts that are of medium complexity. These results are important for messaging platform operators, indicating that using a mix of on-device and cloud resources can help launch AI features for consumers without the usual increase in costs that comes with relying only on the cloud. The architecture also strengthens the privacy posture of the platform by minimizing the volume of raw user text that must leave the device, an advantage that grows in importance as regulatory scrutiny of AI data handling intensifies. Gartner has predicted that edge AI will reach a plateau of productivity within five to ten years [1], and the results presented here suggest that AI-enhanced messaging represents one of the most promising near-term application domains for this paradigm.

5.2 Limitations: Device Constraints and Battery-Aware Routing

The hybrid architecture has limitations. Sustained on-device inference introduces non-trivial energy consumption on client devices, particularly on smartphones where battery life is a primary user concern. Running quantized language models, even small ones, engages processor and memory resources that compete with other foreground and background applications. To address this issue, the architecture suggests a battery-aware routing heuristic that stops local inference when the device's battery level drops below a safe level. Instead, all prompts are sent to the cloud until the device is charged again. Recent studies on battery-aware edge AI for energy-efficient IoT devices have developed smart strategies that control local computing based on the current battery level, creating a solid plan for using this method in messaging systems [9]. Additionally, device heterogeneity presents a challenge: the performance characteristics of on-device inference vary widely across hardware tiers. Tests with SpecEdge on various consumer-grade GPUs, like the RTX 4090, RTX 4070 Ti Super, RTX 3090, and RTX 2080 Ti, show that all setups benefit from faster performance, but the more powerful edge GPUs provide even better The Model Registry and dynamic LoRA loading help manage different devices by allowing the selection of models that fit each device, but more effort is needed to create strong methods for profiling and adapting to these differences.

5.3 Future Directions: Federated Learning and Adaptive Systems

Several promising research directions extend beyond the scope of the current work. Federated

learning provides a means to perpetually enhance on-device SLMs without the centralization of user data on external servers. Federated protocols can improve the accuracy of local models over time by allowing collaborative, privacy-preserving model refinement across distributed device populations. This feature is in line with the same privacy principles that drive the hybrid architecture itself. Research on ways to train small language models on edge devices using federated learning has shown that this teamwork approach can work well even when devices and networks are different, laying the groundwork for adding federated updates to the proposed Model Registry [10]. In addition to federated learning, adaptive thresholding methods that change the Complexity Router's limits based on current cloud usage, device ability, and user choices show a natural improvement in the routing system. The deep gradient compression methods discovered in edge intelligence research can shrink the size of gradients by 270 to 600 times for different model types without losing accuracy, which could make it easier to send federated updates in mobile areas with slow internet. Cross-device model sharing is a new way for nearby devices to work together using peer-to-peer connections to handle requests and lessen reliance on the cloud in areas with many devices.

Table 1: *Complexity Router – Prompt Routing Tiers [1, 2, 3]*

Prompt Complexity	Inference Tier	Processing Location	Example Tasks
Simple	Edge SLM Only	On-Device	Auto-complete, sentiment detection, short summarization
Medium	Speculative Edge Decoding	Edge (draft) + Cloud (verify)	Multi-turn conversation, translation, Q&A
Complex	Cloud LLM Only	Cloud Server	Nuanced multi-turn reasoning, mathematical reasoning, RAG

Table 2: *Security & Privacy Framework Layers [1, 5, 6]*

Security Layer	Location	Mechanism	Purpose
PII Redaction	On-Device (Edge)	Local SLM privacy firewall	Remove names, credit cards, CJIS data before transmission
Guardrail Adapter	On-Device (Edge)	LoRA-based classifier	Detect and block prompt injection/jailbreak attempts
Encrypted Transit	Network	Standard encryption	Protect data during cloud transmission
Trusted Execution Environment	Cloud (Server)	Azure Confidential VMs + NVIDIA H100 GPUs	Encrypted inference; data decrypted only inside TEE enclave

Figure 1: *Speculative Decoding Speedup [3, 7]*

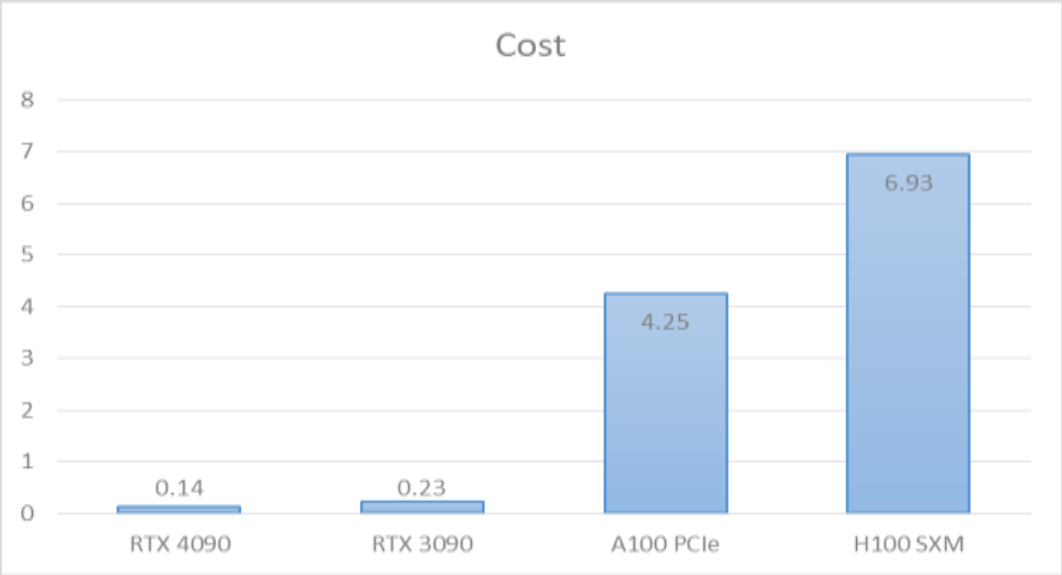

Figure 2: *Edge vs. Server GPU Token Cost [7]*

6. Conclusions

This paper has presented a hybrid-state architecture for AI-enhanced real-time messaging that distributes inference between edge-deployed small language models and cloud-hosted large language models. The architecture is built on three main ideas: a Complexity Router that sorts and directs prompts based on how much thinking they need, a Speculative Edge Decoding protocol that changes cloud processing from making full responses to quickly checking them, and a security system that combines removing personal information on the device with safe processing in Trusted Execution Environments. Tests in simulated busy environments show that this hybrid method significantly lowers cloud costs and response times compared to using only the cloud, while still providing response quality similar to large language models based on standard automated measures. The architecture further strengthens user privacy by ensuring that simple messages never leave the device and that escalated prompts are processed within hardware-isolated enclaves. These results suggest that the best way to improve AI messaging in the future is not just using the cloud but combining it with edge computing, which smartly manages different resources to balance intelligence, cost, speed, and privacy for all user interactions.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.
- **Use of AI Tools:** The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] Zhi Zhou, et al., "Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing," Proceedings of the IEEE (Vol. 107, Issue 8), Aug. 2019. Available: https://ieeexplore.ieee.org/document/8736011

[2] Mozhgan Navardi, et al., "GenAI at the Edge: Comprehensive Survey on Empowering Edge Devices with Generative Artificial Intelligence," arXiv (Preprint for IEEE Surveys), February 2025. Available: https://arxiv.org/html/2502.15816v1

[3] Heming Xia, et al., "Unlocking Efficiency in Large Language Model Inference: A Comprehensive Survey of Speculative Decoding," arXiv (Accepted to ACL Findings 2024), 15 Jan 2024. Available: https://arxiv.org/abs/2401.07851

[4] Ying Shengi, et al., "S-LoRA: Serving Thousands of Concurrent LoRA Adapters," Proceedings of Machine Learning and Systems (MLSys 2024), 6 Nov 2023. Available: https://arxiv.org/abs/2311.03285

[5] Bangyi Yang, "Navigating Privacy Risks in Generative AI: Concerns, Challenges, and Potential Solutions," ResearchGate, February 2025. Available: https://www.researchgate.net/publication/399150042_Navigating_Privacy_Risks_in_Generative_AI_Concerns_Challenges_and_Potential_Solutions

[6] Marcin Chrapek, et al., "Confidential LLM Inference: Performance and Cost Across CPU and GPU TEEs," IEEE International Symposium on Workload Characterization (IISWC), October 2025. Available: https://www.researchgate.net/publication/397822094_Confidential_LLM_Inference_Performance_and_Cost_Across_CPU_and_GPU_TEEs

[7] Jinwoo Park et al.,, "SpecEdge: Scalable Edge-Assisted Serving Framework for Interactive LLMs," arXiv (Distributed Systems), 18 Nov 2025. Available: https://arxiv.org/pdf/2505.17052

[8] Jiangsu Du, et al., "EcoServe: Enabling Cost-effective LLM Serving with Proactive Intra- and Inter-instance Scheduling," arXiv (Cloud Computing Performance), 25 April 2025. Available: https://arxiv.org/html/2504.18154v1

[9] Manne Bhagya Rekha, et al., "Battery-Aware Edge AI For Energy-Efficient Smart Home IoT Devices," International Journal of Creative Research Thoughts (IJCRT), January 2026. Available: https://www.ijcrt.org/papers/IJCRT2601648.pdf

[10] Megan Stewart, et al., "Federated Learning Strategies for Training Small Language Models on Edge Devices," ResearchGate, 2023. Available: https://www.researchgate.net/publication/391156388_Federated_Learning_Strategies_for_Training_Small_Language_Models_on_Edge_Devices