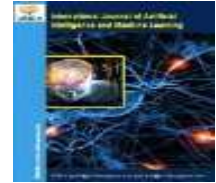




SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

Efficient Reasoning at the Edge: A Systematic Review of Small Reasoning Models, Test-Time Scaling, Quantization, and Energy-Aware AI Inference

Dr. G. Jagan Naik¹, Dr. Burla Srinivas², Dr. A. Ramesh³, Y Amrutha⁴, Dr. Mohammad Shahbaz Khan⁵, Vankudoth Ramesh⁶

¹Associate Professor, Computer Science and Engineering, CMR Institute of Technology, Kandlakoya, Medchal, Hyderabad-501401, gjnaik1106@gmail.com

²Associate Professor, Dept of CSE(AI & ML), CMR Institute of Technology, Kandlakoya, Medchal, Hyderabad -501401, Telangana, India. drburlasrinivas@gmail.com

³Associate professor, Dept of CSE, Sree chaitanya college of engineering, Karimnagar, Telangana 505001, India dr.adicherlaramesh3033@gmail.com

⁴Assistant professor, Malla Reddy Deemed to be University, Hyderabad, Telangana-501401, yellanki.ammu@gmail.com

⁵Associate Professor, Department of Electronics and Communication Engineering, CMR Institute of Technology, Kandlakoya, Medchal, Hyderabad - 501401 shahbazkhan434@gmail.com

⁶Senior Assistant professor, CSE-DS, CVR college of engineering, Hyderabad, Telangana, v.ramesh406@gmail.com

ABSTRACT

Edge devices are increasingly required to have reasoning capabilities and deployed under extreme constraints of memory, latency, energy, network and privacy. In this systematic review, we give an overview of the current research on small reasoning models, test-time scaling, model quantization, hardware-aware optimization and energy-aware edge-cloud inference. Literature was identified by using structured multi-database search and pre-defined eligibility criteria, quality-assessment and data-extraction criteria. The papers analyzed here show that it is possible to obtain valuable reasoning abilities even with much smaller architectures for compact transformers, knowledge distillation, parameter-efficient adaptation, and carefully selected training data. The difficult tasks' performance is further improved by adaptive test-time computation in those more reasoning tokens, more candidate paths and/or verification are provided for the difficult tasks as needed. Post training and quantization-aware methods help to lower bandwidth and memory usage, but could introduce errors to the arithmetic operations and may impact the calibration for multi-step reasoning/inference without any hardware-specific optimizations. Apart from the energy efficient deployment, coordinated voltage and frequency scaling, early exit, workload scheduling, model selection and selective cloud offloading are also important. The compromise that is required between accuracy, speed, memory usage and energy usage makes it impossible to achieve the best performance in all of the above for any one hardware configuration across all application classes in mobile, wearable, robotic, automotive and Internet-of-Things applications. To move forward, there is a need for standardized hardware-level benchmarking, powerful compressed reasoning, adaptation without compromising privacy, explainable decision mechanisms, special accelerators, and self-adaptive systems that can dynamically balance reasoning quality to latency, energy, thermal limits and carbon intensity during actual deployment.

Keywords: Edge artificial intelligence; small reasoning models; test-time scaling; quantization; energy-aware inference; edge-cloud computing; hardware-aware optimization.

1. Introduction

AI has evolved from simple, task-specific algorithms to large language models with the ability to reason, plan, solve math problems, write code, and make complex decisions in multiple steps. These capabilities have been further enhanced by techniques like chain-of-thought prompting, reinforcement learning, self-consistency, verifier-guided search, and test-time computation, but many high-performing reasoning models have billions of parameters and require access to cloud accelerators, high amounts of memory and significant computational power. This is not a suitable environment for such use in resource constrained edge devices like smartphones, embedded systems, wearable devices, autonomous robots, IoT nodes and more. Small reasoning models are thus a crucial step towards making AI more accessible, responsive, and locally deployable. Recently, it has been discovered that various strategies such as tightly filtering training data, designing efficient architectures, knowledge distillation, parameter-efficient adaptation, and reasoning oriented post training can lead to competitive performance when using compact models. MobileLLM introduced the idea of deep-and-thin architectures and embedding sharing, grouped-query attention, and block-wise weight sharing with less than a billion parameters to apply to on-device tasks [1]. Likewise, Phi-3-mini was shown to achieve high language and reasoning performance with a model size of just 3.8 billion parameters that is suitable for deployment on mobile devices [2]. The advantages of these models are reduced cloud dependency, reduced communication latency, support in offline mode, data privacy and real-time decision making. Still, the scaling of edges remains constrained for lack of processing power, memory, battery life, and heat resistance, and for the cost of the long

reasoning chains. One possible solution is test-time scaling: repeating the computation of the inference, either sampling, searching, verifying or adaptive reasoning, at test time. It is known that a compute-allocation strategy carefully designed can lead to a smaller model that will be more effective than a much larger model on specific reasoning problems [3]. But this does create a major balance point on the critical edge between the accuracy of the reasoning, the response time, the memory consumption and the energy consumption. Another option is quantization, where model weights and activations are quantified using less precision. In particular, with activation-aware weight quantization, important weights are preserved, while at the same time, the model size and inference speed are reduced by a small number of bits [4]. Furthermore, the authors evaluated these quantized language models in the real-world scenarios on edge devices, further emphasizing that varying model architectures, quantization levels, devices, and task category can have different energy efficiency, inference latency and accuracy on output [5]. Though some advancements have been achieved, small reasoning model, test-time scaling, quantization, hardware optimization and energy-efficient inference are explored independently. The systematic review is therefore a synthesis of evidence on all aspects related to sustainable private and reliable edge intelligence – including the main methods and assessment measures, accuracy-latency-memory-energy trade-offs and gaps in research and directions for future development of sustainable private and reliable edge intelligence.

2. Systematic Review Methodology

2.1 Research Questions and Review Protocol

The Preferred Reporting Items for Systematic Reviews and Meta-Analyses 2020 statement and the PRISMA-S extension for reporting literature searches [6,7] were used as a framework to design this systematic review. The research objectives, information sources, search terms, eligibility criteria, screening procedure, data-extraction fields, quality assessment and evidence-synthesis strategy were prespecified. The methodology also included suggestions for systematic reviews in computing and software-engineering research [8]. The following research questions were considered in the review: Are there small reasoning models that can be evaluated on resource constrained edge platforms? RQ2: What are the small reasoning models evaluated for resource constrained edge platforms? RQ2: What are the architectural strategies considered and examined with resource constrained edge platforms? RQ2: How will the performance in terms of accuracy, response time, memory consumption and computational cost be affected by test time scaling methods? RQ3: What are the best quantization and model-compression methods for the performance–efficiency trade-off? RQ4: How to measure the power consumption and energy efficiency of different edge devices? RQ5: What are some of the limitations, gaps, and future research needs for efficient reasoning at the edge?

2.2 Literature-Search Strategy and Databases

A comprehensive literature search was designed on the following databases: ScienceDirect, Web of Science Core Collection, IEEE Xplore, ACM Digital Library, arXiv, Scopus and SpringerLink, from the beginning of the database until September 2026. Other discovery methods included backward and forward citation searching, done using Google Scholar. The search strategy employed was a combination of controlled terms and free text words for reasoning models, edge deployment, model optimisation and efficiency outcomes. Among the Boolean expressions, one of them was: (“small reasoning model” OR “small language model” OR “compact language model” OR “on-device language model”) AND (“edge AI” OR “edge computing” OR “mobile device” OR “embedded system” OR “IoT”) AND (“test-time scaling” OR “inference-time compute” OR “quantization” OR “compression” OR “distillation”) AND (energy OR latency OR memory OR efficiency). The query was customized for the syntax and indexing of each database. The names of databases, platforms, search dates, full text search and restrictions were recorded following the recommendations of PRISMA-S for reproducing the results [7].

2.3 Inclusion and Exclusion Criteria

Studies were eligible to be included if they: (i) studied a small, compact & compressed language model; (ii) dealt with deployment on edge, mobile, embedded, wearable, robotic, automotive and/or IoT hardware; (iii) evaluated test-time scaling, quantization, pruning, or knowledge distillation or optimization for hardware and/or energy efficiency; (iv) reported at least one measurable outcome, such as reasoning accuracy, model size, memory usage, latency, throughput, power consumption, or energy per inference; and (v) were published as full-text journal articles, conference papers or as technically detailed preprints in English. Editorials, commentaries, tutorials, patents, inaccessible publications, duplicate publications and studies and investigations, which weren't relevant to the subject and didn't have deployment results, were not included.

2.4 Study Screening, Selection, and Data Extraction

All the retrieved records were first deduplicated and then imported into reference-management software before screening. Titles and abstracts were independently judged by two reviewers, and full text review of potentially eligible reports. Problems were addressed and, when necessary, a third reviewer consulted with the parties. The reference lists and citation of articles were checked for other studies. There were 600 database records and 94 records from supplementary methods in the provisional PRISMA dataset. 446 records in the database were screened after eliminating 146 duplicates and eight records for other reasons. Of these 79 (63%) were evaluated for eligibility and 7 were not available. Of those, 136 full text reports were requested, 7 were not available, 129 were assessed for eligibility and 79 (63%) were eligible. These were supplemented by

another nine reports, bringing the total to 90 reports that were evaluated after four of these were unavailable. Following all the full text exclusions 53 studies remained for provisional inclusion. This is a selection pathway and reason for exclusion as outlined in Figure 1. Bibliographic information, the model of the family, the number of parameters, the reasoning task, optimization method, quantization accuracy, hardware architecture, benchmark data set, baseline comparator, accuracy, latency, throughput, memory footprint, power consumption, energy per inference, and reported limitations were extracted using a standardized template.

2.5 Quality Assessment and Synthesis Approach

The methodological quality was assessed with an adapted technical checklist, which includes the clarity of the goals, completeness of model and data descriptions, suitability of the baselines, transparency of hardware and software configuration, reproducibility of the implementation, completeness of efficiency measures, statistical validity and disclosure of the limitations. Yes = 2 and No = 0 and partly = 1. The percentage of the maximum score was used to determine the quality of studies in the following way: high quality (75–100%), moderate quality (50–75%), and low quality (<50%). The results were summarized narratively, and presented in structured, comparative tables rather than in statistical meta analysis, because of the anticipated high diversity of model architectures, reasoning benchmarks, quantization specifications, hardware platforms and energy measurement procedure. Where possible it was compared with respect to accuracy change, compression ratio, tokens per second, peak memory, energy per inference and accuracy per watt. The process of complete identification, screening, eligibility and inclusion was reported following the official PRISMA 2020 flow-diagram framework.

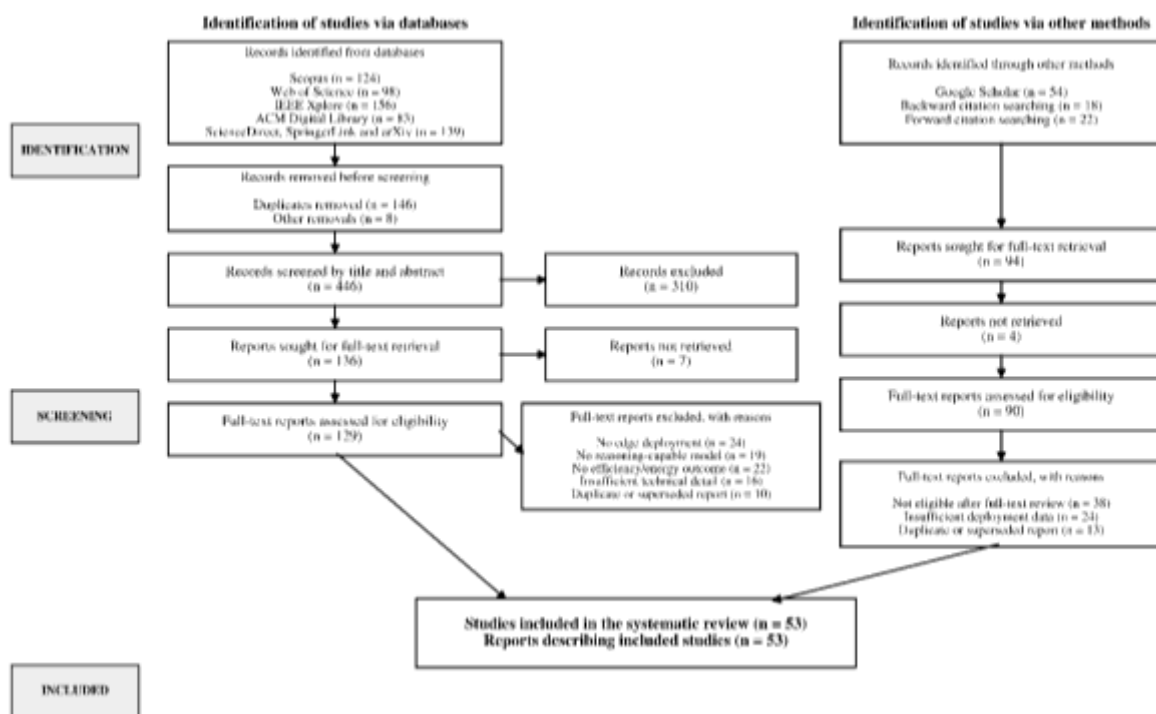


Figure 1. PRISMA Flow Diagram of the Literature Search and Study-Selection Process

3. Small Reasoning Models for Edge Intelligence

Small reasoning models are small generative models designed to be used in instruction following, decomposition of tasks, answering questions, selecting tools, and deciding on actions in context, when memory, processing and energy requirements are limited, e.g., in embedded systems, single-board computers, smartphones, wearables and robots. The general architecture is the Transformer but efforts have been made to enhance the computational efficiency by reducing the number of parameters, layer-wise parameter sharing, attention grouped across several queries, sharing of weights, compact embeddings, the optimized attention kernel, and the well-designed training data. TinyLlama has 1.1B parameters, is pretrained with ~1 trillion tokens, is based on the Llama 2 architecture and FlashAttention as well as other open-sourced optimizations, and, importantly, downstream performance is competitive with a relatively small compute cost [6]. Additionally, OpenELM offers layer-wise scaling, so that fewer pretraining tokens are used while keeping the same number of parameters: the model with ~1 billion parameters outperforms OLMo with half the number of pretraining tokens by 2.36% and the support to convert OpenELM models to Apple's MLX framework allows for inference and fine-tuning on consumer devices [7]. The more recent MobileLLM-Pro incorporates four-bit quantization-aware self-distillation, implicit positional distillation, specialist-model merging and utility-guided data mixing into a one-billion-parameter model that is capable of supporting context windows up to 128,000 tokens and which is said to experience only slight performance drop following four-bit quantization [8]. A

comparison of the architectural properties, reasoning ability, deployability and major drawbacks of representative small models is given in the table below. Apart from designing compact models at the outset, further compression techniques, such as knowledge distillation, structured pruning, low rank decomposition, embedding compression, and quantization can be employed to further decrease the resource requirements of the models. It achieved about 2-fold in embedding compression for language task performance similar to the original models with a reduction in energy consumption per query of around 50% on Raspberry Pi hardware using pretrained token embeddings for GPT-2 variants with the help of the Tensor-Train Decomposition. Additionally, edge models can be specialized without saving or updating the entire model weights as part of parameter-efficient adaptation. QLoRA optimizes memory usage by freezing the base model to four bits, training low-rank adapters, and using low-rank adapters, in addition to NormalFloat4, double quantization and paged optimizers to keep the memory cost for adaptation as low as possible and still obtain performance close to that of traditional 16-bit fine-tuning [10]. The strategies can be adapted to local medical assistance, interpretation of industrial faults, mobile tool calling, task planning in the robot, educational assistance, smart home control and privacy-preserving customized services. It is true for large models, but recent testing of small models for edge-based agentic tasks indicates that the models with 1-3 billion parameters do better in multiple-tool, parallel-function, and multi-step interactions than do ultra-compact models with less than 1 billion parameters [11]. Model selection should not be based on the number of parameters, therefore. The accuracy of the reasoning, task completion, first-token latency, generation speed, peak memory, energy per query, thermal stability, and robustness after compression should be evaluated in practice. Finally, to address these risks, small models may have to be used to obtain a fuller plan, but may still be able to produce a fuller plan, may fail to select the appropriate tools, or may be able to produce plausible but incorrect plans; reliable edge systems should therefore be supported by small models with restricted decoding, local retrieval, confidence estimation, rule based validation, safety filters, and escalation to an edge server or cloud model if some uncertainty or complexity in the task exceeds a threshold.

Table 1. Comparative Characteristics of Small Reasoning Models for Edge Deployment

Model family	Year	Parameters	Principal architectural or training characteristics	Reported reasoning evidence	Edge-deployment relevance	Main limitation
DistilBERT	2019	66M	Transformer encoder compressed through knowledge distillation during pretraining	Retained approximately 97% of BERT's language-understanding capability	Approximately 40% smaller and 60% faster than BERT; suitable for classification and extraction	Encoder-only design does not support general autoregressive reasoning
MobileLLM	2024	125M/350M	Deep-and-thin architecture, embedding sharing and grouped-query attention	Accuracy improvements of 2.7/4.3 percentage points over earlier corresponding baselines; competitive API-calling performance	Specifically optimized for sub-billion-parameter mobile inference	Limited knowledge capacity and weaker complex reasoning than larger models
Phi-3-mini	2024	3.8B	Decoder-only Transformer trained on filtered public and synthetic data; instruction and safety alignment	69% MMLU and 8.38 MT-Bench	Demonstrated feasibility of deployment on a mobile phone	Memory, latency and energy still depend strongly on precision and context length
Gemma 2B	2024	2B	Lightweight decoder-only open model derived from Gemini research; pretrained and instruction-tuned checkpoints	Gemma family showed strong language-understanding and reasoning performance and outperformed	Compact open model suitable for local adaptation and quantized inference	The original report provides limited device-level energy measurements

				similarly sized open models on many evaluated tasks		
Phi-3-small	2024	7B	Scaled Phi-3 architecture trained on 4.8 trillion tokens	75% MMLU and 8.7 MT-Bench	Greater reasoning capacity for high-end edge devices and edge servers	Higher memory and computational requirements than sub-4B models
Gemma 7B	2024	7B	Decoder-only open model with pretrained and instruction-tuned variants	Competitive performance across language understanding, reasoning and safety benchmarks	Suitable for edge servers or accelerated devices after compression	Typically requires aggressive quantization for memory-constrained mobile deployment
Phi-3-medium	2024	14B	Larger Phi-3 family member trained on 4.8 trillion tokens	78% MMLU and 8.9 MT-Bench	Useful as an edge-server model or escalation model in hierarchical inference	Generally unsuitable for resource-constrained endpoints without substantial compression

4. Test-Time Scaling and Adaptive Reasoning

Test-time scaling involves scaling (allocating) more computation resources when the model is used for inference instead of increasing the number of model parameters or retraining the model. The model can produce longer reasoning traces, explore various possible solutions, check intermediate solutions, or revise an initial answer in an iterative fashion. A Chain-of-thought prompting technique, which involves breaking down problems into a series of steps, has proven to be useful for this purpose, enhancing arithmetic, symbolic and commonsense reasoning [18]. Self-consistency does the same but samples a variety of reasoning paths and then chooses the result that is most agreed upon, albeit at the cost of higher energy consumption and latency if the results are repeated [19]. There are iterative approaches that allow the model to give feedback and correct its own results without training, such as Self-Refine, which also leads to an overhead for each iteration [20]. In the case of edge deployment, the adaptive controllers should send their reasoning depending on the difficulty of the task, their confidence, battery status, thermal state, and latency requirement. A light-weight first-pass response will be marked based on confidence and/or verification signals as illustrated in Figure 2; easy tasks or high confidence tasks will be able to go out the door, and uncertain tasks will be assigned more reasoning tokens, alternative solution paths or step-level verification. The second method of defect identification is process supervision where one may be able to identify the errors in intermediate operations before they get to the final output [21]. This adaptive strategy will allow us to achieve a practical balance between accuracy, response time, memory and energy requirements such that simple tasks will not need to be a victim of having to stretch the reasoning uniformly.

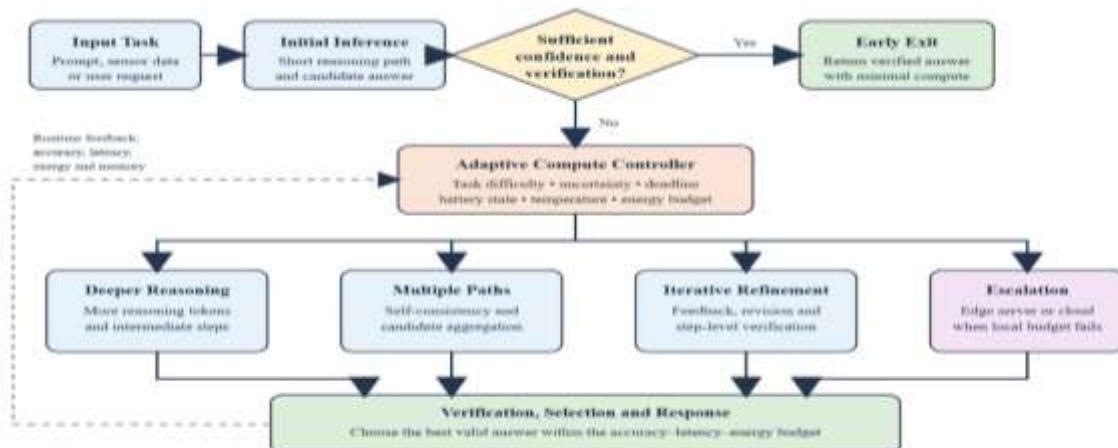


Figure 2. Test-Time Scaling Framework for Adaptive Reasoning at the Edge

5. Quantization and Hardware-Aware Model Optimization

This quantization enables the storage, communication, computation and inference of the model with reduced precision for model weights, activations and key-value caches, and consequently reduced storage requirements, memory traffic, computational load, and if implemented using optimized kernels, reduced inference speed and energy use. Post training quantization is the conversion of an already trained model with a relatively small calibration set, and thus it is cost efficient and practical to be deployed on the edge. ZeroQuant uses fine-grained, hardware-compatible weight/activation quantization, layer-wise knowledge distillation and an optimized inference backend that led to minimal INT8 weight/activation accuracy loss, up to $5.19\times$ inference speeds with INT8 weights and activations, and a $3\times$ memory reduction under a mixed INT4/INT8 configuration [22]. Indeed, OmniQuant extends low-bit PTQ with learnable weight clipping and equivalent transformations that limit activation outliers, applicable to configurations of any size, from W6A6 to W2A16 [23]. Conversely, quantization-aware training introduces noise during training to make the model parameters robust to quantization noise. Usually this maintains accuracy more at a very low level of precision, but it will need more training data, memory and computation amounts. The authors of BitNet v2 present a training strategy that relies on one bit weights and Hadamard-transformed four bit activations that reduces the number of computations and memory requirements for inference with only a minor degradation in performance [24]. One such "mixed" precision approach is to maintain a higher precision for the most important weights and/or operations while aggressively quantizing the less important ones. For example, in SpQR, the outliers' weights have been stored with higher precision, and the weights which have not changed too much have been reduced to three or four bits, which led to over fourfold compression of the weights' memory without affecting the performance loss within the assessed models. The compression algorithm AQLM improves this to approximately 2 to 3 bits per parameter by imposing additive codebooks, and by optimizing block-level coding for CPU and GPU to use, for memory constrained inference [26]. But, model-based reasoning can be especially sensitive to errors in the low bits of the numbers since the errors can sum up along the route of reasoning, affect token probabilities, or cause intermediate numbers to be off. Apart from perplexity, Quantized models should be assessed for their accuracy in reasoning, calibration, consistent output and task completion. There isn't a single best quantization method: INT8 is very much a safe choice for deployment; INT4 is a good memory/accuracy tradeoff; and the sub-3-bit methods give the most benefit from compression, but depend on a small number of high-accuracy values being stored, require some special kernels and need careful calibration. The quantization format, group size, operator fusion, memory layout and accumulation precision must then be optimized to be hardware-aware, and consider the characteristics of the target CPU, GPU, NPU, or digital-signal processor. If the arithmetic supported by the accelerator computer does not implement a low bit arithmetic for the small model, or if quantization/ dequantization operations are repeated with excessive overhead, a small model may have relatively high latency or energy consumption.

Table 2. Comparison of Quantization Methods for Edge-Based Reasoning Models

Method	Category	Typical precision	Main optimization principle	Reported efficiency benefit	Expected effect on reasoning accuracy	Hardware suitability	Main limitation
ZeroQuant	PTQ with optional layer-wise distillation	W8A8 or mixed W4/W8A8	Fine-grained hardware-friendly quantization and optimized backend	Up to $5.19\times$ speedup; mixed configuration provided approximately $3\times$ memory reduction	INT8 generally preserves accuracy; mixed INT4 may require distillation	GPUs and accelerators supporting INT8/INT4 kernels	Backend-specific optimization is required to realize speedup
OmniQuant	Calibrated PTQ	W6A6, W4A4, W4A16, W3A16 or W2A16	Learnable weight clipping and equivalent transformations for activation outliers	Low-cost calibration with memory and real-device inference benefits	Better preservation than naïve rounding at low precision, but degradation rises near two bits	GPU, CPU and edge accelerators with corresponding low-bit kernels	Calibration and block-wise optimization add preparation time

SpQR	Sparse mixed-precision PTQ	Mostly 3–4-bit weights with sparse higher-precision outliers	Isolates sensitive outlier weights and quantizes remaining weights	More than 4× memory compression and reported 15% speedup for a 33B model on a consumer GPU	Near-lossless perplexity in evaluated LLaMA and Falcon models	Consumer GPUs; potentially adaptable to edge GPUs	Sparse metadata and irregular access may reduce efficiency on some accelerators
AQLM	Additive weight-only PTQ	Approximately 2–3 bits per parameter	Represents weights through sums of learned codebook vectors	Very small memory footprint; optimized CPU and GPU token-generation implementations	Strong size–accuracy trade-off below three bits, but reasoning tasks require separate validation	Memory-limited CPUs and GPUs	Codebook lookup and reconstruction require specialized kernels
AdaDim	Outlier-aware weight-only PTQ	Sub-4-bit weights with higher-precision activations	Selects input- or output-channel quantization according to weight sensitivity	Improves existing low-bit PTQ without changing the base model	Reported gains on MMLU and HumanEval relative to conventional grouping	Small-batch and mobile-oriented inference	Benefits depend on model-specific sensitivity patterns
BitNet v2	Quantization-aware/native low-bit training	1-bit weights and 4-bit activations	Hadamard transformation suppresses activation outliers before quantization	Reduced memory and computational cost for batched inference	Minimal degradation on reported relative to its one-bit baseline	Future low-bit accelerators and specialized kernels	

6. Energy-Aware AI Inference at the Edge

Abovementioned is the goal of energy-aware edge inference of minimizing environmental and operational impact of reasoning with acceptable accuracy and response time. Energy per query (joules per token produced), power, battery depletion, latency, throughput, and CO₂ equivalent of electricity use and carbon intensity should be reported. Carbontracker demonstrates the value of measuring performance, energy and carbon emissions [28]. Dynamic voltage and frequency scaling (DVFS) scales the CPU, GPU, and memory clocks on the fly to meet the demands of the workload, but a low frequency can cause the computer to execute for longer periods, which can result in consuming more energy. Mobile-LLM measurements reveal that independently-configured governors can result in sub-optimal frequency combinations, which has led to a need for coordinated control between the processing units [29]. There are also options such as choosing a smaller model, restricting reasoning tokens, using early exit, batching appropriate requests or deferring non-emergent tasks. Content-aware early exiting combined with reported energy savings of 26% on average for edge video inference [30] is a combination of content-aware early exiting and is called E4. When computation cannot be handled locally, it can be partitioned and/or offloaded to cloud to meet the required accuracy, latency, memory or energy requirements through Edge–cloud collaboration. DVFO optimizes both the frequencies of the devices as well as offloading decisions together and reduced energy usage by an average of 33% with only a small loss of accuracy [31]. The energy-aware controller continuously evaluates task complexity, model confidence, battery state, thermal conditions, network quality, privacy requirements and reasoning budget, and determines

which model to use, how stringent it needs to be, how much reasoning it can do, which frequency hardware it should use and where it should be run. The measured accuracy, latency, memory and energy are then used in the next scheduling decision, enabling an adaptive and carbon-aware reasoning over a variety of edge and cloud resources.

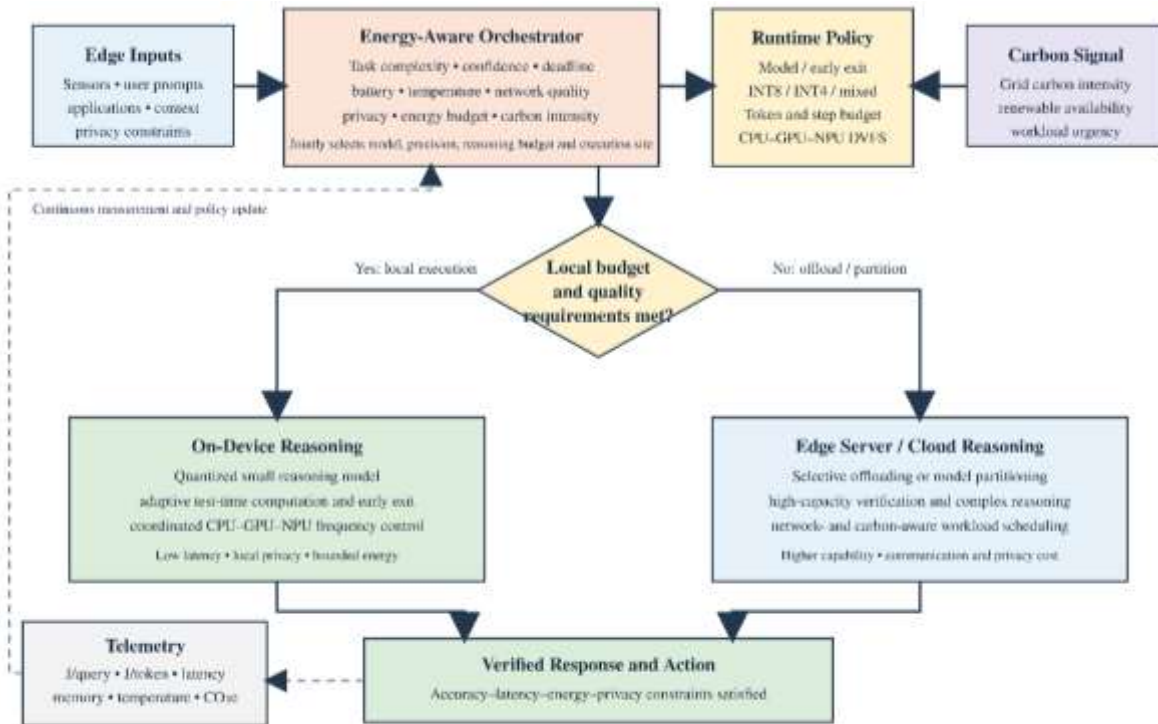


Figure 3. Integrated Architecture for Energy-Aware Reasoning and Edge-Cloud Inference

7. Performance Evaluation and Application Domains

Efficient edge reasoning performance evaluation should be based on both task and deployment metrics. The accuracy, F1 score, exact match, calibration rate, tasks completed, time to first token, inter-token latency, throughput, peak memory, model size, energy per query and joules per token are used for measuring utility; whereas time to first token, inter-token latency, throughput, peak memory, model size, energy per query and joules per token are used for measuring operational efficiency. For smartphone platforms, heterogeneous CPU, GPU, NPU chips should be considered in the evaluation, because the performance of the chipsets could be different in different frameworks in a smartphone [32]. The MLPerf Tiny includes representative, low latency, energy efficient workloads for microcontroller and IoT nodes for keyword spotting, visual wake word, image classification and anomaly detection [33]. The reduced sampling rates have demonstrated significant decreases in RAM, latency, energy and operations while maintaining reasonable accuracy, particularly for time-series processing, UCIHAR, WISDM, PAMAP2 or MHEALTH. In addition, the success of the planning process, deadlines of control cycles, rate of recovery, safety violations, thermal throttling and the ability to sustain performance through continuous use of sensor streams should be considered for robotic and automotive applications [34]. Mobile assistants highlight response quality and first-token latency, wearables highlight energy and memory, robots and vehicles highlight deterministic latency and reliability and industrial IoT highlights availability, diagnostic accuracy and energy efficiency [35]. Between comparisons, the prompts, precision, context size, batch size, software stack, ambient conditions and power measurement boundaries should be the same to avoid drawing incorrect conclusions.

Table 3. Benchmarking of Efficient Reasoning Approaches Across Edge-AI Applications

Application	Example task	Efficient approach	Primary metrics
Mobile	Assistant and tool selection	Quantization and early exit	Accuracy, first-token latency, RAM, I/query
Wearable	Health and activity interpretation	Reduced sampling and compact models	F1 score, battery use, latency, memory
Robotics	Planning and fault recovery	Adaptive reasoning and verification	Task success, control latency, reliability
Automotive	Driver assistance and scene reasoning	Edge-cloud collaboration	Accuracy, deadline compliance, energy
Industrial	Fault diagnosis and	TinyML and model	Diagnostic accuracy,

IoT	maintenance	selection	availability, J/inference
-----	-------------	-----------	---------------------------

8. Research Challenges and Future Directions

While the compressed reasoning model has proven to be successful, it is still very important to take into account the calibration and verification in the face of numerical errors, distribution shifts, adversarial prompts and confidence estimates. The extraction of models, the injection of prompts and leakage of data from the context are yet to be addressed security concerns when executing locally [36,37]. Explainability needs to show evidence of the decision made but not the traces of any sensitive reasoning. Limits of benchmarking include inconsistent prompts, hardware, precision, power boundaries and short duration tests and continuous deployment can lead to degradation effects not seen in a static benchmark, such as thermal and temporal degradation [38]. Future research efforts should focus on accelerators with neuromorphic arrays and special low-bit accelerators, on a universal energy measurement of the models, and on a hardware–software co-design and self-adaptive controller that would allow a trade-off between accuracy, latency, privacy, and energy [39].

Conclusion

The key is to reason efficiently at the edge, which demands coordination of optimization across model architecture, test-time computation, quantization, hardware control and edge–cloud scheduling. Distributing additional computation when there is uncertainty or complexity of the task can be achieved through adaptive reasoning, while small models can provide private, responsive intelligence. The quantification will not only save memory and bandwidth, it will also have to be tested on real machines for use in reasoning tasks to make sure it is reliable. The energy-aware scheduling should include the battery state, network quality and carbon intensity. Future systems will include local models, special accelerators, verification modules and selective escalation to the cloud. While benchmarking has already been a fundamental part of the laboratory workflow, reproducible application-specific benchmarking is still very much the backbone of the movement towards dependable and sustainable edge intelligence.

References

1. Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*. 2021;372:n71. doi:10.1136/bmj.n71
2. Kitchenham B, Charters S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*. Keele University and Durham University; 2007. Report No.: EBSE-2007-01.
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. *Proc IEEE*. 2019;107(8):1738–1762. doi:10.1109/JPROC.2019.2918951
4. Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: vision and challenges. *IEEE Internet Things J*. 2016;3(5):637–646. doi:10.1109/JIOT.2016.2579198
5. Liu Z, Zhao C, Iandola F, Lai C, Tian Y, Fedorov I, et al. MobileLLM: optimizing sub-billion parameter language models for on-device use cases. 2024. arXiv:2402.14905
6. Abidin M, Jacobs SA, Awan AA, Aneja J, Awadallah A, Awadalla H, et al. Phi-3 technical report: a highly capable language model locally on your phone. 2024. arXiv:2404.14219
7. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. 2015. arXiv:1503.02531
8. Han S, Mao H, Dally WJ. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. In: *Proceedings of the International Conference on Learning Representations*; 2016. Paper
9. Hu EJ, Shen Y, Wallis P, Allen-Zhu Z, Li Y, Wang S, et al. LoRA: low-rank adaptation of large language models. In: *Proceedings of the International Conference on Learning Representations*; 2022. Paper
10. Wei J, Wang X, Schuurmans D, Bosma M, Xia F, Chi E, et al. Chain-of-thought prompting elicits reasoning in large language models. In: *Advances in Neural Information Processing Systems*. 2022;35:24824–24837. Paper
11. Wang X, Wei J, Schuurmans D, Le QV, Chi EH, Narang S, et al. Self-consistency improves chain of thought reasoning in language models. In: *Proceedings of the International Conference on Learning Representations*; 2023. Paper
12. Snell C, Lee J, Xu K, Kumar A. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. 2024. arXiv:2408.03314
13. Yao S, Zhao J, Yu D, Du N, Shafran I, Narasimhan K, Cao Y. ReAct: synergizing reasoning and acting in language models. In: *Proceedings of the International Conference on Learning Representations*; 2023. Paper
14. Graves A. Adaptive computation time for recurrent neural networks. 2016. arXiv:1603.08983
15. Teerapittayanon S, McDanel B, Kung HT. BranchyNet: fast inference via early exiting from deep neural networks. In: *Proceedings of the 23rd International Conference on Pattern Recognition*. IEEE; 2016. p. 2464–2469. doi:10.1109/ICPR.2016.7900006

16. Frantar E, Ashkboos S, Hoefler T, Alistarh D. GPTQ: accurate post-training quantization for generative pre-trained transformers. In: *Proceedings of the International Conference on Learning Representations*; 2023. Paper
17. Lin J, Tang J, Tang H, Yang S, Chen W-M, Wang W-C, et al. AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. *Proc Mach Learn Syst*. 2024;6:87–100. Paper
18. Dettmers T, Lewis M, Belkada Y, Zettlemoyer L. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In: *Advances in Neural Information Processing Systems*. 2022;35:30318–30332. Paper
19. Xiao G, Lin J, Seznec M, Wu H, Demouth J, Han S. SmoothQuant: accurate and efficient post-training quantization for large language models. In: *Proceedings of the 40th International Conference on Machine Learning*. 2023;202:38087–38099. Paper
20. Dettmers T, Pagnoni A, Holtzman A, Zettlemoyer L. QLoRA: efficient finetuning of quantized LLMs. In: *Advances in Neural Information Processing Systems*. 2023;36:10088–10115. Paper
21. Ma S, Wang H, Ma L, Wang L, Wang W, Huang S, et al. The era of 1-bit LLMs: all large language models are in 1.58 bits. 2024. arXiv:2402.17764
22. Wang K, Liu Z, Lin Y, Lin J, Han S. HAQ: hardware-aware automated quantization with mixed precision. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019. p. 8612–8620. Paper
23. Sze V, Chen YH, Yang TJ, Emer JS. Efficient processing of deep neural networks: a tutorial and survey. *Proc IEEE*. 2017;105(12):2295–2329. doi:10.1109/JPROC.2017.2761740
24. Schwartz R, Dodge J, Smith NA, Etzioni O. Green AI. *Commun ACM*. 2020;63(12):54–63. doi:10.1145/3381831
25. Strubell E, Ganesh A, McCallum A. Energy and policy considerations for deep learning in NLP. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence: ACL; 2019. p. 3645–3650. doi:10.18653/v1/P19-1355
26. Henderson P, Hu J, Romoff J, Brunskill E, Jurafsky D, Pineau J. Towards the systematic reporting of the energy and carbon footprints of machine learning. *J Mach Learn Res*. 2020;21(248):1–43. Paper
27. Li E, Zeng L, Zhou Z, Chen X. Edge AI: on-demand accelerating deep neural network inference via edge computing. *IEEE Trans Wireless Commun*. 2020;19(1):447–457. doi:10.1109/TWC.2019.2946140
28. Kang Y, Hauswald J, Gao C, Rovinski A, Mudge T, Mars J, Tang L. Neurosurgeon: collaborative intelligence between the cloud and mobile edge. In: *Proceedings of the 22nd International Conference on Architectural Support for Programming Languages and Operating Systems*. 2017. p. 615–629. doi:10.1145/3037697.3037698
29. Li E, Zhou Z, Chen X. Edge intelligence: on-demand deep learning model co-inference with device-edge synergy. In: *Proceedings of the 2018 Workshop on Mobile Edge Communications*. 2018. p. 31–36. doi:10.1145/3229556.3229558
30. Reddi VJ, Cheng C, Kanter D, Mattson P, Schmuelling G, Wu C-J, et al. MLPerf inference benchmark. In: *Proceedings of the 47th Annual International Symposium on Computer Architecture*. 2020. p. 446–459. doi:10.1109/ISCA45697.2020.00045
31. Ignatov A, Timofte R, Kulik A, Yang S, Wang K, Baum F, et al. AI Benchmark: running deep neural networks on Android smartphones. In: *European Conference on Computer Vision Workshops*. Cham: Springer; 2019. p. 288–314. doi:10.1007/978-3-030-11021-5_19
32. Banbury CR, Reddi VJ, Torelli P, Holleman J, Jeffries N, Kiraly C, et al. MLPerf Tiny benchmark. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. 2021;1. Paper
33. Davies M, Srinivasa N, Lin TH, Chinya G, Cao Y, Choday SH, et al. Loihi: a neuromorphic manycore processor with on-chip learning. *IEEE Micro*. 2018;38(1):82–99. doi:10.1109/MM.2018.112130359
34. Mehonic A, Kenyon AJ. Brain-inspired computing needs a master plan. *Nature*. 2022;604:255–260. doi:10.1038/s41586-021-04362-w
35. Jagan Naik G, Burla S, Vijendar A, Rajyalaxmi J, Madhu B, Sridhar K. Intelligent medical devices and healthcare management: advances in mechanical engineering and artificial intelligence. *Int J Comput Inf Syst Ind Manag Appl*. 2026;18(20s):290–311. doi:10.70917/ijcisim-2026-5181.
36. Preethi B, Chethan GS, Sheikh N, Jagan Naik G, Chidambaram S, Kalit D. Development and evaluation of an explainable AI-driven decision support system for clinical and operational healthcare management. *J Intell Decis Mak Inf Sci*. 2026;3(7s):2230–2244. ISSN: 3079-0875.
37. Jagan Naik G, Korra S, Yedukondalu G, Dash P, Gugulothu P, Biksham V. A multimodal artificial intelligence framework for information integration and intelligent decision support.
38. Jagan Naik G, Silpa Raj M, Rajarao B, Balaram A, Jyothsna K, Ramesh V. Intelligent data analytics using explainable and interpretable machine learning techniques. *Int J Artif Intell Mach Learn*. 2026;6(10s).
39. Zhang X, Wang Y, Shi W. pCAMP: performance comparison of machine learning packages on the edges. In: *USENIX Workshop on Hot Topics in Edge Computing*. Boston: USENIX Association; 2018. Paper
40. Xu D, Li T, Li Y, Su X, Tarkoma S, Jiang T, et al. Edge intelligence: empowering intelligence to the edge of network. *Proc IEEE*. 2021;109(11):1778–1837. doi:10.1109/JPROC.2021.3119950