Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Toward Sustainable On-Device Intelligence: A Survey on Energy-Efficient RAG Systems with Small Language Models

Zhiyuan Cheng

zhycheng@stanford.edu

Stanford University https://orcid.org/0009-0003-2845-5322

Longying Lai

University of Rochester https://orcid.org/0009-0000-3515-450X

Yue Liu

Rutgers, The State University of New Jersey

Yu Sun

University of Pennsylvania

Research Article

Keywords: On-device AI, Retrieval-Augmented Generation, Small Language Models, Neural Processing Unit, Quantization, Green AI, Energy Efficiency, Sustainable Computing, Edge Intelligence

Posted Date: August 7th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10597421/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Toward Sustainable On-Device Intelligence: A Survey on Energy-Efficient RAG Systems with Small Language Models

Zhiyuan Cheng[a,*], Longying Lai[b], Yue Liu[c] and Yu Sun[d]

[a]School of Engineering, Stanford University, Stanford, CA, USA
[b]Simon Business School, University of Rochester, Rochester, NY, USA
[c]Accounting & Information Systems, Rutgers University, Newark, NJ, USA
[d]School of Engineering and Applied Science, University of Pennsylvania, Philadelphia, PA, USA

ARTICLE INFO

Keywords:
On-device AI
Retrieval-Augmented Generation
Small Language Models
Neural Processing Unit
Quantization
Green AI
Energy Efficiency
Sustainable Computing
Edge Intelligence

ABSTRACT

The proliferation of Large Language Models (LLMs) has driven a paradigm shift toward on-device inference, motivated by privacy preservation, latency reduction, and offline capability. Simultaneously, Retrieval-Augmented Generation (RAG) has emerged as the dominant pattern for grounding language model outputs in external knowledge. However, deploying RAG pipelines on resource-constrained edge devices—smartphones, IoT gateways, and embedded systems equipped with Neural Processing Units (NPUs)—introduces a complex multi-dimensional trade-off among accuracy, latency, memory footprint, energy consumption, and carbon emissions. While each pillar has been surveyed individually, the three-way intersection of *on-device AI inference optimization*, *retrieval-augmented generation*, and *Green AI / sustainability* has not previously been drawn together into a unified system-design perspective, and this survey undertakes that cross-domain synthesis. We systematically examine (i) model compression techniques—quantization, pruning, knowledge distillation—with emphasis on NPU-aware optimization; (ii) on-device RAG system architectures including lightweight retrieval, context compression, and modular frameworks; and (iii) energy measurement, carbon footprint estimation, and sustainable inference strategies for edge deployment. As our core contribution, we propose ALEMC, an evaluation framework extending conventional metrics with explicit *Energy* and *Carbon* dimensions, and synthesize design principles for building sustainable on-device RAG systems. We review 142 papers spanning 2017–2026, identify open challenges in hardware–software co-design, standardized energy benchmarking, and lifecycle-aware model selection, and outline future research directions at this rapidly evolving intersection.

1. Introduction

Large Language Models (LLMs) have fundamentally transformed natural language processing, achieving remarkable capabilities in reasoning, generation, and instruction following. Yet this progress has come at an escalating computational cost. Training GPT-3 consumed an estimated 1,287 MWh of electricity and emitted approximately 552 tonnes of CO_2—the equivalent of 123 gasoline-powered passenger vehicles driven for one year [8]. Inference at scale dwarfs even these figures: recent estimates suggest that serving a widely deployed conversational AI system consumes more energy daily than training the underlying model did in its entirety [120]. At the same time, the industry is racing to move AI capabilities to edge devices—smartphones, wearables, IoT gateways, and embedded systems—driven by demands for privacy preservation, low latency, offline operation, and reduced cloud dependency. This juxtaposition between the unsustainable trajectory of ever-larger models and the resource-constrained reality of edge hardware defines the central tension motivating this survey.

We argue that three independently maturing research threads are now converging in a way that demands unified analysis. Each addresses a critical dimension of the challenge, but no existing framework integrates all three into a coherent systems perspective. This survey bridges that gap by treating on-device Retrieval-Augmented Generation (RAG) as a *sustainability-aware system design problem*—one in which model architecture, retrieval pipeline design, hardware constraints, and environmental impact must be co-optimized.

Pillar 1: On-device AI and Small Language Models. The emergence of Neural Processing Units (NPUs) capable of 22–75 TOPS within mobile power envelopes has made local inference not merely feasible but increasingly practical. Concurrently, a new generation of Small Language Models (SLMs)—ranging from roughly 100M to 7B parameters—has demonstrated that carefully designed architectures trained on curated data can rival much larger models on targeted tasks. MobileLLM [89] operates within 2 GB memory budgets on smartphones; TinyLlama [140] shows that extended training on 3 trillion tokens can compensate for reduced parameter count; and Gemma [42] provides open-weight models with strong multilingual capability at the 2B–7B scale. When combined with aggressive quantization techniques such as GPTQ [37] and AWQ [83], these models can be compressed to 3–4 bits per weight with negligible accuracy loss, fitting comfortably within the memory and compute constraints of modern mobile hardware.

*Corresponding author
zhycheng@stanford.edu (Z. Cheng)
ORCID(s): 0009-0003-2845-5322 (Z. Cheng)

Pillar 2: Retrieval-Augmented Generation. RAG has emerged as the dominant paradigm for grounding language model outputs in external knowledge, mitigating hallucination, and enabling knowledge updates without costly retraining. Lewis *et al.* [73] formalized the approach as a joint retrieval-generation architecture, and dense retrieval methods such as DPR [69] have since established strong baselines for open-domain question answering. Beyond these well-known benefits, RAG carries an underappreciated sustainability advantage: by retrieving only the passages most relevant to a query, it allows the generator to operate on a compact, targeted context rather than the long prompts demanded by pure long-context inference. Because prefill compute scales linearly—and for standard attention, quadratically—with context length, and KV cache energy during decoding scales with the cached prefix, precisely scoped retrieval directly reduces per-query energy consumption. In this sense, RAG is not merely a grounding technique but a *computational parsimony* mechanism that replaces indiscriminate long-context processing with evidence-targeted context delivery. However, the standard RAG pipeline assumes cloud-scale resources: large embedding models, GPU-backed vector indices, and generous memory budgets for long-context generation. Translating this paradigm to edge devices requires rethinking every component. Recent systems such as MiniRAG [33] and EdgeRAG [107] have begun this translation, demonstrating that on-device RAG is achievable through lightweight embeddings, compressed vector indices, and selective context retrieval—though at the cost of increased design complexity and nuanced accuracy–efficiency trade-offs.

Pillar 3: Green AI and Sustainability. A growing body of research has documented the environmental cost of the AI scaling paradigm. Strubell *et al.* [111] showed that training a single large Transformer model can emit as much CO_2 as five automobiles over their lifetimes, while Schwartz *et al.* [106] introduced the influential Red AI / Green AI distinction, advocating that efficiency should be reported alongside accuracy as a first-class metric. Carbon tracking tools such as CodeCarbon [23] and inference-specific estimators like CO2-Meter [40] now make it possible to measure the environmental footprint of AI workloads at per-query granularity. Sustainability concerns also extend beyond operational energy to upstream material supply chains: He *et al.* [51] document that AI data center scaling is increasingly constrained by rare-earth element availability, with cost and schedule risks that compound the carbon considerations and motivate a shift toward smaller, edge-deployed systems that demand fewer specialized hardware accelerators per deployed query. Yet sustainability considerations remain largely disconnected from the on-device AI and RAG design processes: models are compressed for speed and memory, not for carbon; retrieval pipelines are optimized for recall, not for energy.

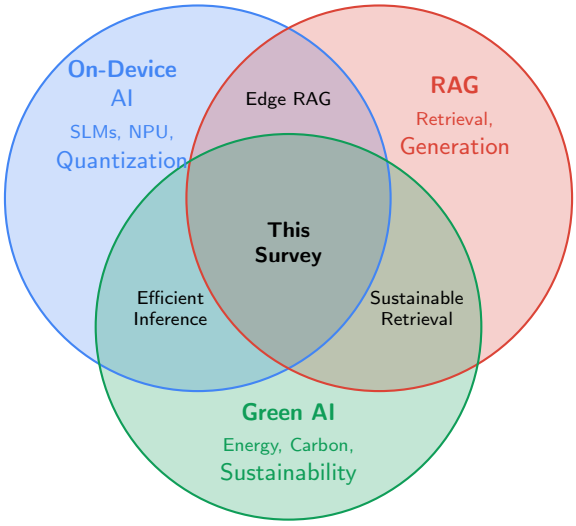


Figure 1: Scope of this survey at the three-way intersection of on-device AI, retrieval-augmented generation, and Green AI / sustainability.

1.1. The Gap

Each of the three pillars is already served by mature surveys, but each covers at most *two* of the three dimensions. Comprehensive reviews of on-device language model deployment—Xu *et al.* [132], Zheng *et al.* [143], the SLM survey of Wang *et al.* [121], the model-compression-and-systems survey of Chen *et al.* [12], and the edge-intelligence survey of Wang *et al.* [125]—address hardware constraints, compression, and (in part) inference-time energy efficiency, but overlook RAG integration. These build in turn on broad LLM surveys [141, 65] and dedicated treatments of efficient architectures [116], model compression [145], and efficient inference for reasoning models [87] that establish the single-pillar foundations our synthesis draws together. RAG surveys such as Gao *et al.* [41] and Sharma [108] provide thorough coverage of retrieval strategies, evaluation frameworks, and failure modes, yet do not consider on-device deployment or sustainability. Systematic reviews of Green AI—Verdecchia *et al.* [119], Rojahn and Grum [104], and the carbon-aware AI taxonomy of Devadas [29]—map energy measurement and lifecycle models but rarely examine RAG pipelines or NPU-specific optimization. What no prior survey does is treat the three-way intersection where sustainable on-device RAG systems reside as a single design space. That intersection is, in fact, remarkably sparse: only a handful of papers sit squarely within it—carbon-footprint estimators for edge LLMs (CO2-Meter [40], EnviroLLM [3]), energy-aware serving and reporting (GreenLLM [86], AI-CARE [1]), and carbon-aware adaptation (Carbon-Aware LoRA Scaling [142], GaiaFlow [39])—and most are 2025–2026 preprints that address individual components rather than the integrated system. It is precisely *because* this literature is nascent and fragmented that a synthesis is valuable: the contribution of this work is not to catalogue a large existing body of intersection research but to unify three adjacent communities into a coherent framework before their

conventions diverge. This absence is problematic because the design decisions are deeply interdependent: the choice of quantization bit-width affects both NPU throughput *and* energy consumption per token; the retrieval strategy determines both answer quality *and* the computational overhead of context processing; and the deployment location—edge versus cloud—shifts not only latency and privacy characteristics but also the carbon equation from operational to embodied emissions.

1.2. Contributions

This survey makes the following contributions:

1. **A unifying synthesis at the three-way intersection.** Building on the mature single- and dual-pillar surveys of on-device LLMs [132, 143, 121], RAG [41, 108], and Green AI [119, 104], we provide what is, to our knowledge, the first review to jointly cover on-device AI inference, retrieval-augmented generation, and Green AI / sustainability as a single design space. Because the genuinely three-way intersection literature is still small [40, 3, 86, 1, 142, 39], our contribution is a synthesis—analyzing interdependencies among design choices that no prior single- or dual-pillar survey captures—rather than a catalogue of an established intersection field.
2. **Systematic taxonomy of 142 papers.** We classify 142 papers published between 2017 and 2026 into nine categories spanning foundation models, SLM architectures, quantization and compression, NPU hardware, RAG fundamentals, on-device RAG systems, Green AI measurement, evaluation frameworks, and application case studies.
3. **ALEMC framework.** We propose ALEMC—*Accuracy, Latency, Energy, Memory, Carbon*—an evaluation framework that extends conventional performance metrics with explicit Energy and Carbon dimensions, providing a principled basis for comparing on-device RAG configurations across the full sustainability spectrum.
4. **On-device RAG integration patterns.** We synthesize three design patterns: *quantized SLM with lightweight retriever*, *hybrid edge–cloud fallback*, and *carbon-aware adaptive pipeline*. They occupy distinct points in the trade-off space among accuracy, efficiency, and sustainability.
5. **Open challenges and research roadmap.** We identify six open challenges, spanning NPU hardware gaps, standardized energy benchmarks, lifecycle-aware model selection, federated on-device RAG, multi-modal edge retrieval, and regulatory implications. We outline future research directions for each.

1.3. Paper Organization

Fig. 1 illustrates the scope of this survey as the intersection of three research domains. The remainder of this paper is organized as follows. Section 2 describes our systematic review methodology, including the search strategy, selection criteria, and classification taxonomy. Section 3 provides background on Transformer-based language models, the evolution toward SLMs, the RAG paradigm, Green AI fundamentals, and the NPU hardware landscape. Section 4 reviews model compression techniques—quantization, pruning, knowledge distillation, and inference optimization—with emphasis on NPU compatibility. Section 5 surveys on-device RAG system architectures, efficiency techniques, and failure modes. Section 6 examines energy measurement frameworks, sustainable inference strategies, and the carbon comparison between on-device and cloud deployment. Section 7 presents the core synthesis: the ALEMC evaluation framework, design space taxonomy, integration patterns, and case studies for sustainable on-device RAG. Section 8 identifies open challenges and future research directions. Section 9 concludes the survey.

2. Review Methodology

This survey follows a structured literature review approach inspired by the PRISMA 2020 guidelines [98].[1] We describe our search strategy, selection criteria, and classification taxonomy below.

2.1. Search Strategy

We conducted systematic searches across five academic databases: IEEE Xplore, ACM Digital Library, Scopus, Google Scholar, and arXiv. The initial search was performed between January and March 2026, followed by a targeted citation-chaining update through August 2026, covering publications from 2017 to August 2026. We constructed Boolean search queries combining terms from three domains:

- **On-device AI:** “on-device LLM” OR “edge inference” OR “small language model” OR “mobile AI” OR “NPU” OR “neural processing unit” OR “model compression” OR “quantization”
- **RAG:** “retrieval-augmented generation” OR “RAG” OR “retrieval-augmented” OR “on-device retrieval” OR “edge RAG” OR “vector search”
- **Green AI:** “green AI” OR “sustainable AI” OR “energy-efficient inference” OR “carbon footprint” OR “energy consumption AI” OR “carbon-aware computing”

Queries were formed as pairwise AND combinations of cross-domain terms (*e.g.*, “on-device LLM” AND “energy efficiency”) as well as the three-way conjunction to specifically target the intersection.

[1] The PRISMA 2020 statement [98] is cited here as the methodological guideline for reporting the review and is *not* part of the reviewed corpus; it is therefore excluded from the 142-paper count and from the classification table in the Appendix.

2.2. Inclusion and Exclusion Criteria

Inclusion criteria: (i) Peer-reviewed journal articles, conference papers, or well-cited arXiv preprints; (ii) publications addressing at least one of the three domains; (iii) English-language publications; (iv) papers presenting novel methods, systems, benchmarks, or comprehensive surveys.

Exclusion criteria: (i) Workshop abstracts or extended abstracts without technical depth; (ii) duplicate publications across venues (only the most complete version retained); (iii) purely theoretical papers without empirical validation or system design relevance; (iv) papers on cloud-only LLM serving without edge or sustainability considerations.

2.3. Selection Process

The initial search yielded 1,247 candidate papers across the five databases. After removing 395 duplicates, 852 unique records remained. Title and abstract screening against the inclusion criteria reduced this set to 281 papers, which were subjected to full-text review. Of these, 173 were excluded for insufficient technical depth, lack of empirical validation, insufficient relevance to the three-way intersection, or scope limited to cloud-only deployment without edge or sustainability relevance, yielding a final corpus of 108 papers from the systematic search; 34 further papers identified through backward and forward citation chaining were subsequently included, bringing the total corpus to 142 papers analysed in this survey. Fig. 2 illustrates the complete selection funnel.

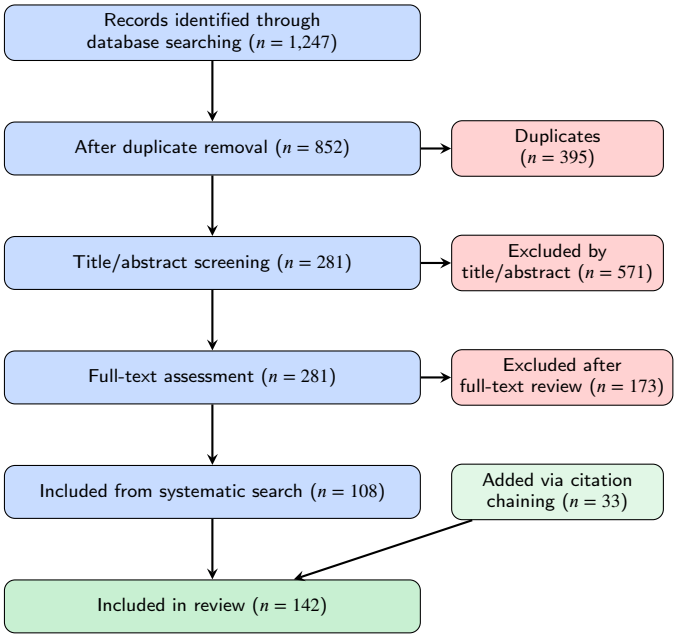


Figure 2: PRISMA-style literature selection process.

Corpus versus bibliography. Because the two counts differ, we state the relationship explicitly. The *systematic corpus* comprises the 142 screened papers, and these—and only these—are enumerated in the classification table of the Appendix and counted in every statistic reported in this section. The *bibliography* contains 147 entries: the 142 corpus papers plus five items that were never subject to the screening protocol. Two are not research literature about the surveyed topic, namely the PRISMA 2020 reporting guideline [98], which defines our method, and the `llama.cpp` inference runtime [43], which is software used in the empirical study of Section 7.3. The remaining three are foundational financial question-answering references—the FinQA [15] and TAT-QA [144] benchmarks and the hierarchical-retrieval system of Choe *et al.* [22]—cited in the finance case study (Section 7.4) to ground that application in independent, peer-reviewed evidence; they fall outside our search queries, which targeted the three-way on-device/RAG/sustainability intersection rather than general-domain financial QA, and are therefore excluded from the corpus. Any reference cited in the body of this survey is thus either a corpus paper or one of these five clearly identified exceptions.

2.4. Literature Classification

We classify the reviewed literature into nine categories organized around the three pillars of this survey:

A. Foundation & Background (8 papers): Transformer architecture, scaling laws, foundational LLM papers

B. Small Language Models (14 papers): SLM architectures, distilled models, mobile-optimized LMs, task-specific efficiency

C. Quantization & Compression (23 papers): PTQ, QAT, pruning, low-rank adaptation, NPU-aware techniques, distillation, dataset efficiency

D. NPU & Hardware (8 papers): NPU architectures, inference engines, hardware benchmarks, NPU co-design

E. RAG Core (20 papers): RAG fundamentals, dense/sparse retrieval, vector databases, RAG surveys, hybrid and multi-modal retrieval, conflict and evidence calibration

F. On-Device RAG (19 papers): Edge RAG systems, mobile RAG, context pruning, query routing, privacy-preserving RAG, agentic memory, and multi-modal retrieval

G. Green AI & Sustainability (19 papers): Energy measurement, carbon footprint, sustainable inference, carbon-frugal search, carbon-aware scheduling, greenwashing oversight, supply-chain sustainability

H. Evaluation & Benchmarks (14 papers): RAG evaluation, LLM benchmarks, embedding and quantization benchmarks, robust evaluation, bias evaluation, unlearning audits, unified metric taxonomies

I. Applications & Case Studies (17 papers): Healthcare, smart home, IoT, finance, security, domain-specific and multi-modal AI

Fig. 3 presents the hierarchical taxonomy of reviewed literature, and Table 13 in the Appendix maps each paper to its category, year, and venue.

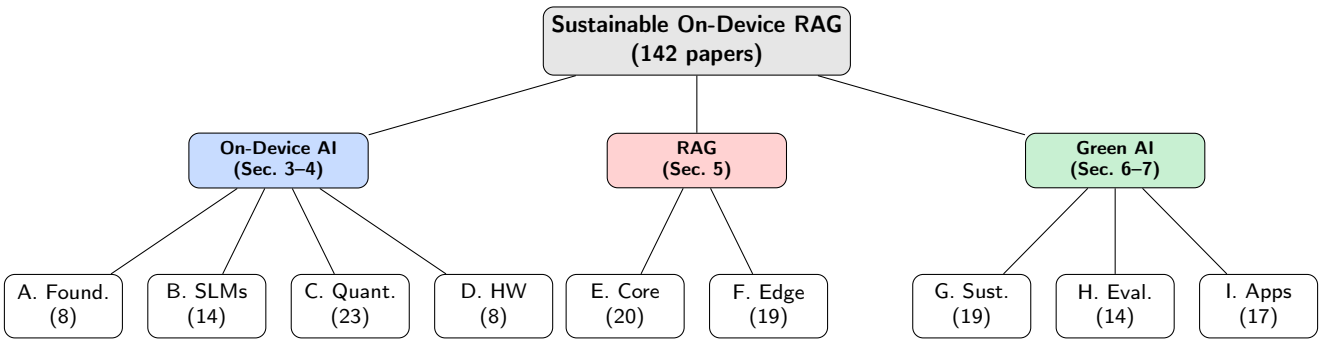


Figure 3: Hierarchical taxonomy of the 142 reviewed papers organized by the three pillars and nine categories.

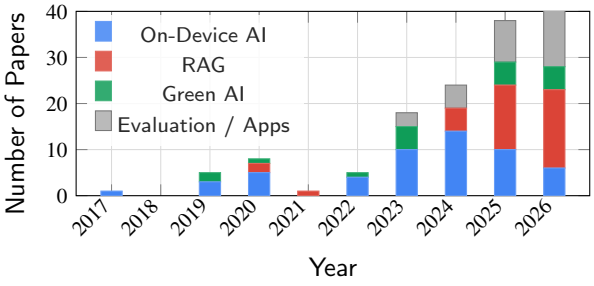


Figure 4: Temporal distribution of the 142 reviewed papers by primary domain. The sharp acceleration from 2023 onward reflects the rapid maturation and convergence of on-device AI, RAG, and Green AI research.

2.5. Bibliometric Overview

The temporal distribution of reviewed papers, shown in Fig. 4, reveals a sharp acceleration: only 11% of papers (15 of 142) were published before 2022, while 73% (104 of 142) appeared in 2024–2026, reflecting the rapid convergence of on-device AI, RAG, and sustainability research. The most represented venues include arXiv (preprints), NeurIPS, ICML, ICLR, ACL, AAAI, EMNLP, ACM Computing Surveys, and IEEE Transactions.

3. Background and Preliminaries

This section establishes the foundational concepts underlying on-device RAG systems. We trace the evolution from large to small language models, introduce the RAG paradigm, formalize the Green AI perspective, and survey the on-device hardware landscape that makes edge deployment feasible.

3.1. From Large Language Models to Small Language Models

The Transformer architecture, introduced by Vaswani *et al.* [118], replaced recurrence with multi-head self-attention, enabling parallelized sequence modeling and establishing the foundation for virtually all modern language models. In the self-attention mechanism, each token computes attention weights over all other tokens in the sequence, yielding a context-aware representation at $O(n^2 d)$ computational cost for sequence length n and model dimension d. This architectural paradigm proved remarkably scalable and triggered a rapid increase in model size.

Kaplan *et al.* [68] formalized this trend through neural scaling laws, demonstrating that cross-entropy loss follows a power-law relationship with model parameters, dataset size, and compute budget. These findings implied that larger models would yield predictably better performance, fueling what Schwartz *et al.* later termed the "Red AI" trajectory [106]. A sequence of landmark models validated this hypothesis: BERT [30] introduced bidirectional pre-training with 340M parameters; GPT-3 [8] demonstrated emergent few-shot capabilities at 175B parameters; and LLaMA [117] showed that open-weight models trained on more data could match larger proprietary systems.

However, deploying models with tens or hundreds of billions of parameters on edge devices is impractical due to memory, power, and latency constraints. This reality has driven the emergence of *Small Language Models* (SLMs)—typically ranging from 100M to 7B parameters—designed to preserve useful language capabilities while fitting within the computational envelope of smartphones, tablets, and embedded systems. The SLM design philosophy inverts the scaling-law argument: rather than maximizing parameters, it seeks to maximize *capability per parameter* through architectural innovation, curated training data, and efficient optimization.

Several SLM families have emerged with distinct strategies. **MobileBERT** [114] pioneered mobile-optimized language models by compressing BERT through bottleneck structures and progressive knowledge transfer, achieving a 4.3× size reduction while retaining 99.2% of BERT-base performance on GLUE. **DistilBERT** [105] demonstrated that task-agnostic knowledge distillation could produce models 60% smaller and 60% faster than BERT while preserving 97% of its language understanding. The **Phi** series [44] from Microsoft challenged the assumption that performance requires scale, showing that models as small as 1.3B parameters trained on carefully curated "textbook-quality" data can outperform models 10× their size on reasoning benchmarks. **TinyLlama** [140] adapted the LLaMA architecture to 1.1B parameters and trained on 3 trillion tokens, demonstrating that small decoder-only

Table 1
Comparison of representative Small Language Models (SLMs) discussed in this survey.

Model	Parameters	Architecture	Target Platform	Year	Key Innovation
MobileBERT [114]	25M	Encoder (BERT)	Mobile / Edge	2020	Bottleneck structures with progressive knowledge transfer from $\text{BERT}_{\text{LARGE}}$
DistilBERT [105]	66M	Encoder (BERT)	General / Edge	2019	Task-agnostic distillation; 60% smaller, 60% faster than BERT
Phi-1.5/2 [44]	1.3–2.7B	Decoder (GPT)	Desktop / Mobile	2023	Textbook-quality data curation; outperforms 10× larger models on reasoning
TinyLlama [140]	1.1B	Decoder (LLaMA)	Edge / Embedded	2024	3T token training; demonstrates scaling-data for small models
Gemma [42]	2B / 7B	Decoder (Gemini)	General / Mobile	2024	Open-weight; strong multilingual and instruction-following performance
Mistral 7B [57]	7.3B	Decoder (custom)	Desktop / Edge	2023	Grouped-query attention + sliding window attention for efficient inference
MobileLLM [89]	125M–1B	Decoder (custom)	Mobile ($\leq$2 GB)	2024	Deep-and-thin architecture; embedding sharing; sub-billion on-device focus

models benefit disproportionately from extended training. **Gemma** [42] from Google provided open-weight 2B and 7B models with strong multilingual and instruction-following capabilities. **Mistral 7B** [57] introduced grouped-query attention and sliding window attention to achieve inference efficiency rivaling models twice its size. **MobileLLM** [89] specifically targeted sub-billion parameter regimes, employing deep-and-thin architectures with embedding sharing and grouped-query attention to operate within 2 GB memory budgets typical of mobile devices.

Recent empirical evidence further validates the SLM paradigm. A task-specific efficiency analysis [10] demonstrates that SLMs can outperform LLMs on focused tasks when properly fine-tuned, challenging the assumption that larger models are universally superior. CoDES [53] proposes a context-efficient framework that enhances SLM capabilities through structured context distillation, enabling compact models to handle complex inputs without increasing parameter count. These findings reinforce that task-specific optimization and context-aware design can compensate for reduced model scale—a principle directly applicable to on-device RAG, where the generator must produce grounded responses within tight resource budgets.

Table 1 summarizes the key characteristics of representative SLM families discussed in this survey. Collectively, these models demonstrate that the path to on-device intelligence lies not in brute-force scaling but in the co-optimization of architecture, data curation, and training strategy—a principle that aligns naturally with the sustainability objectives motivating this review.

3.2. Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) augments language models with an external knowledge retrieval step, enabling the model to condition its output on retrieved evidence rather than relying solely on parametric knowledge encoded during pre-training. Lewis *et al.* [73] formalized the RAG paradigm as a sequence-to-sequence model in which a non-parametric retrieval component and a parametric generator are jointly optimized. The canonical RAG pipeline consists of three stages: *indexing*, *retrieval*, and *generation*.

In the **indexing** stage, a document corpus is partitioned into passages, each encoded into a dense vector representation using an embedding model, and stored in a vector index. In the **retrieval** stage, a user query is similarly encoded and the top-k most similar passages are retrieved via approximate nearest-neighbor search. In the **generation** stage, the retrieved passages are concatenated with the query as context and fed to a language model, which produces a grounded response.

Retrieval strategies broadly fall into two categories. *Dense retrieval* methods learn continuous vector representations that capture semantic similarity. Dense Passage Retrieval (DPR) [69] trains separate BERT-based encoders for queries and passages using contrastive learning, achieving substantial improvements over traditional methods on open-domain question answering. *Sparse retrieval* methods, exemplified by BM25, rely on lexical term matching with TF-IDF weighting and remain competitive baselines due to their efficiency and interpretability. In practice, hybrid approaches that combine dense and sparse scores through reciprocal rank fusion or learned reranking often achieve the best retrieval quality.

Efficient vector storage and search are critical for RAG performance, particularly under on-device constraints. FAISS [64] provides GPU-accelerated and CPU-optimized implementations of approximate nearest-neighbor algorithms including inverted file indices (IVF) with product quantization, enabling billion-scale similarity search with sub-millisecond latency. Lighter-weight alternatives such as sqlite-vss embed vector search within SQLite databases, trading some recall for minimal memory overhead—a compelling property for mobile deployment.

RAG offers several advantages over purely parametric approaches. First, it *reduces hallucination* by grounding generation in retrieved evidence, improving factual accuracy on knowledge-intensive tasks. Second, it enables *knowledge updates without retraining*: the retrieval corpus can be modified without touching model weights, dramatically reducing the cost of keeping systems current. Third, it provides *attribution*: retrieved passages serve as citations, improving the transparency and trustworthiness of generated outputs.

However, RAG introduces its own evaluation challenges. The quality of the end-to-end system depends on both retrieval recall and generation faithfulness, and degradation in either component can propagate to the final output. Furthermore, context window limitations create a retrieval–generation tension: retrieving more passages improves recall but can overwhelm the generator, and models have been shown to disproportionately attend to the beginning and end of long contexts while neglecting the middle [85]. Evaluation frameworks such as RAGAS [32] attempt to decompose RAG quality into component metrics including context relevance, answer faithfulness, and answer relevance, but standardized benchmarks for on-device RAG remain nascent.

3.3. Green AI and Sustainability

The rapid scaling of deep learning has been accompanied by a commensurate increase in energy consumption and carbon emissions. Strubell *et al.* [111] quantified this impact in a widely cited analysis showing that training a single large Transformer model with neural architecture search can emit approximately 284 tonnes of carbon dioxide equivalent (tCO_2e)—comparable to the lifetime emissions of five gasoline-powered automobiles, including their manufacture. This finding catalyzed a broader reckoning with the environmental costs of AI research.

Schwartz *et al.* [106] introduced the influential distinction between *Red AI* and *Green AI*. Red AI prioritizes accuracy improvements at any computational cost, typically reporting only task performance while ignoring the energy required to achieve it. Green AI, by contrast, treats computational efficiency as a first-class evaluation criterion alongside accuracy, advocating that research contributions should be measured in terms of performance *per unit of computation*. This framing has been widely adopted and provides the philosophical grounding for the sustainability dimension of this survey.

The environmental impact of AI systems can be decomposed into two components. **Operational carbon** arises from the electricity consumed during training and inference. It is a function of the hardware power draw, workload duration, and the carbon intensity of the electricity grid, typically expressed in grams of CO_2 equivalent per kilowatt-hour (gCO_2e/kWh). **Embodied carbon** encompasses the emissions from manufacturing, transporting, and eventually disposing of hardware. While operational carbon dominates for large training runs, embodied carbon becomes proportionally more significant for edge devices with short replacement cycles and low per-device utilization.

Verdecchia *et al.* [119] conducted a systematic review of Green AI, identifying lifecycle models that categorize environmental impacts across the full model lifecycle: data collection and preprocessing, training, fine-tuning, inference, and hardware end-of-life. Their analysis reveals that most existing Green AI research focuses narrowly on training-phase energy, while inference-phase and hardware-lifecycle impacts remain underexplored—a gap that is particularly relevant for on-device deployment, where inference dominates the energy budget.

Several key metrics have been established for quantifying the sustainability of AI systems:

- **Energy consumption (kWh):** Total electrical energy consumed during training or inference, measured via hardware power monitors or software estimators such as CodeCarbon [23], the ML CO_2 Impact calculator [71], and Zeus [135].
- **Carbon emissions (gCO_2e):** Energy consumption multiplied by the grid carbon intensity; represents the direct climate impact. Geographical and temporal variation in grid carbon intensity introduces significant variability [31].
- **Power Usage Effectiveness (PUE):** The ratio of total facility power to IT equipment power, capturing data center overhead. Typical PUE values range from 1.1 (hyperscale) to 1.6 (enterprise). For on-device inference, PUE is effectively 1.0.
- **Computational efficiency (FLOPS/watt):** The ratio of floating-point operations per second to power consumption, providing a hardware-normalized measure of energy efficiency. Modern NPUs achieve 5–40 TOPS/watt, compared to 0.5–2 TOPS/watt for mobile CPUs.

This survey argues that a meaningful assessment of on-device AI sustainability requires tracking energy and carbon across the full lifecycle—from model training and compression through deployment-time inference—rather than optimizing any single metric in isolation.

3.4. NPU and On-Device Hardware Landscape

Neural Processing Units (NPUs) are specialized accelerators designed to execute deep neural network operations with high throughput and low power consumption. Unlike general-purpose CPUs and GPUs, NPUs employ dense arrays of multiply-accumulate (MAC) units optimized for low-precision matrix arithmetic (INT8, INT4, and increasingly sub-4-bit formats), achieving high computational throughput within tight power envelopes—typically 1–10 watts for mobile SoCs.

The NPU architecture, illustrated schematically in Fig. 5, typically comprises several key components: (i) a MAC array that performs the bulk of tensor computations; (ii) an on-chip SRAM buffer hierarchy designed to minimize expensive off-chip memory accesses; (iii) a DMA engine for

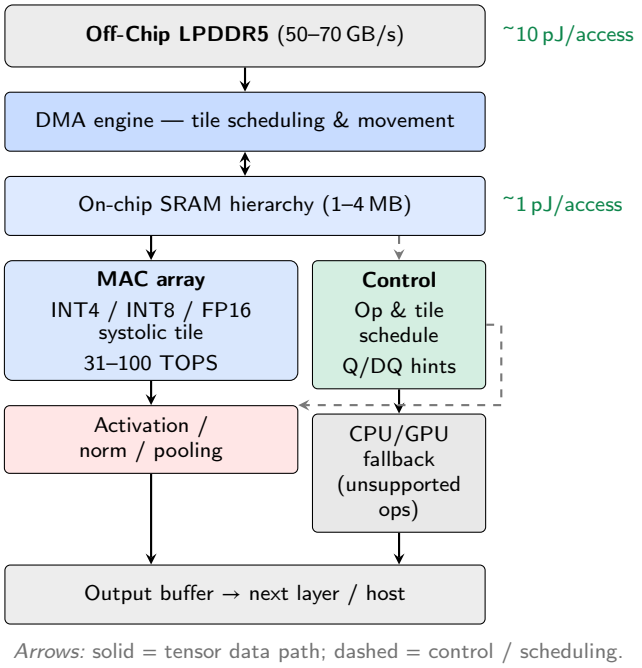


Figure 5: Schematic of a mobile NPU architecture. The MAC array executes low-precision tensor operations at high throughput; the on-chip SRAM hierarchy minimizes expensive off-chip DRAM accesses; unsupported operations fall back to CPU/GPU execution. Energy annotations highlight the order-of-magnitude cost difference between SRAM and DRAM access.

efficient data movement between main memory and the accelerator; and (iv) a control unit that schedules operators and manages data tiling. The efficiency advantage of NPUs stems from exploiting the regular, data-parallel structure of neural network computations: by executing fixed-precision dot products in a systolic or spatial array, NPUs avoid the overhead of instruction fetch, branch prediction, and cache coherence that dominates general-purpose processors.

Several NPU families have achieved widespread deployment in commercial mobile platforms. **Qualcomm Hexagon NPU**, integrated into Snapdragon SoCs, offers 75–100 TOPS in current-generation chipsets (Snapdragon 8 Elite and its 2025 successor) with support for INT8, INT4, FP8, and mixed-precision execution. Its compiler toolchain (Qualcomm AI Engine) supports ONNX and TensorFlow Lite model formats. **Apple Neural Engine (ANE)**, embedded in A-series and M-series chips, provides up to 38 TOPS (M5) with tight integration into the Apple Core ML framework, enabling seamless model deployment across iOS and macOS devices. **MediaTek APU**, found in Dimensity-series SoCs, achieves up to 57 TOPS (Dimensity 9400+) with dedicated integer, FP8, and floating-point processing elements supporting MoE and DiT model architectures. **Samsung Exynos NPU** and **Google Tensor TPU** (in Pixel devices) represent additional mobile NPU implementations with 31–59 TOPS capabilities in 2025 flagship devices.

These NPUs deliver 10–40× greater energy efficiency than CPUs for neural network inference [56], making them indispensable for on-device AI workloads. However, NPU deployment introduces several constraints that directly impact model and system design. **Memory bandwidth** is often the primary bottleneck: mobile LPDDR5 provides 50–70 GB/s, which limits the achievable throughput for memory-bound operations such as autoregressive token generation. **Precision support** varies across NPU families; while INT8 is universally supported, INT4 and sub-4-bit quantization require specific hardware generations or software emulation. **Operator coverage** remains incomplete—NPUs typically accelerate a subset of neural network operations (convolutions, matrix multiplications, activation functions), while unsupported operators fall back to CPU or GPU execution, introducing pipeline stalls and data transfer overhead. Table 2 summarizes the specifications of major mobile NPU families.

Benchmarking NPU performance for language model workloads remains challenging. The AI Benchmark suite [56] evaluates neural network inference across mobile devices but focuses primarily on computer vision tasks. MLPerf Inference [103] provides standardized benchmarks for datacenter and edge scenarios, including language model workloads, but mobile NPU submissions remain sparse. As SLMs become primary targets for NPU deployment, there is a pressing need for benchmarks that specifically measure language model inference throughput, latency, and energy consumption on mobile NPU hardware—a gap we revisit in Section 8.

A promising direction for closing the hardware–model gap is NPU-native co-design, where the model architecture itself is shaped by NPU constraints rather than adapted post hoc. AutoNeural [13] exemplifies this approach by replacing the standard Vision Transformer encoder with a MobileNet-style backbone that maintains bounded activation distributions for stable INT4/8/16 quantization, and pairing it with a hybrid Transformer–State-Space-Model language backbone that eliminates KV cache overhead through gated convolutions. Deployed on a Qualcomm SA8295P NPU, AutoNeural achieves up to 14× lower end-to-end latency and 7× lower quantization error compared to conventional ViT-Transformer baselines, demonstrating that rethinking model topology for NPU constraints yields far greater efficiency gains than post-training compression of GPU-first architectures.

4. Model Compression for On-Device Deployment

Deploying language models on edge devices demands aggressive reduction of model size, memory bandwidth, and arithmetic complexity while preserving task accuracy.

Table 2
Specifications of major mobile NPU families relevant to on-device language model inference.

Vendor	SoC	NPU	TOPS	Precision	Year	Notes
Qualcomm	Snapdragon 8 Elite	Hexagon NPU	75	INT4/INT8/FP16	2024	ONNX/TFLite support; GPU shared memory
Qualcomm	Snapdragon 8 Elite Gen 2	Hexagon NPU	100*	INT4/INT8/FP16/FP8	2025	~2× predecessor TOPS; prelim. specs
Apple	A18 Pro	ANE	35†	INT8/FP16	2024	16-core ANE; iPhone 16 Pro; Apple Intelligence
Apple	M5	ANE	38†	INT8/FP16	2025	Up to 128 GB unified memory; high-BW ANE link
MediaTek	Dimensity 9400+	APU 890	57*	INT4/INT8/FP16/FP8	2025	MoE/DiT/LoRA support; NeuroPilot SDK
Samsung	Exynos 2500	NPU	59	INT8/FP16	2025	24K MACs; 3nm GAA; +39% over Exynos 2400
Google	Tensor G5	Edge TPU	31*	INT8/BF16	2025	TSMC 3nm; +60% TPU vs. G4; Pixel 10

All TOPS figures are peak vendor-quoted throughput from product announcements and marketing materials, not independent measurements; peak TOPS does not translate directly into sustained language-model throughput, which is typically memory-bandwidth-bound. *Estimated from official performance-improvement announcements (*e.g.*, "~2× the prior generation"); exact figures not yet formally disclosed. †Apple does not publish official ANE TOPS; values are widely cited third-party estimates and should be treated as approximate.

This section reviews four complementary families of compression and optimization techniques—quantization, pruning, low-rank adaptation and distillation, and inference-level optimizations—with particular attention to their compatibility with the Neural Processing Unit (NPU) datapaths introduced in Section 3. The techniques surveyed here correspond primarily to taxonomy category C (Quantization & Compression, 23 papers), drawing additionally on categories B (Small Language Models) and D (NPU & Hardware); the full per-paper classification is given in Table 13 (Appendix).

4.1. Quantization

Quantization reduces the numerical precision of model weights and activations from the native 16- or 32-bit floating-point representation to lower bit-widths (*e.g.*, 8-bit integer, 4-bit integer, or even ternary), thereby shrinking model size and enabling the use of integer arithmetic units that dominate mobile NPU silicon. We organize quantization methods along three axes: *when* quantization is applied (post-training vs. during training), *what* is quantized (weights only vs. weights and activations), and *how* bit-widths are assigned (uniform vs. mixed-precision).

4.1.1. Post-Training Quantization (PTQ)

PTQ methods convert a pre-trained floating-point model to a lower-precision representation without any additional training, making them attractive for on-device deployment where retraining budgets are prohibitive.

LLM.int8(). Dettmers *et al.* [27] observe that a small fraction of hidden-state dimensions—so-called *emergent outliers*—carry magnitudes orders of magnitude larger than the remaining channels. Naively applying 8-bit quantization to these outlier-containing matrices introduces severe accuracy degradation. LLM.int8() addresses this through a *mixed-precision decomposition*: outlier dimensions (typically fewer than 0.1% of columns) are retained in 16-bit floating point, while all remaining dimensions are quantized to INT8 and multiplied using efficient integer matrix kernels. The approach enables inference of models with up to 175 B parameters with negligible perplexity increase, and crucially demonstrates that quantization difficulty concentrates in a predictable, sparse subset of dimensions—an insight that motivates subsequent activation-aware methods.

GPTQ. Frantar *et al.* [37] formulate weight quantization as a layer-wise reconstruction problem and solve it via an approximate second-order method derived from Optimal Brain Quantization. GPTQ processes each weight column sequentially, quantizing it to the target precision and then updating all not-yet-quantized columns to compensate for the induced error using the inverse Hessian of the layer's squared reconstruction loss. The method operates in a single forward pass over a small calibration set (typically 128 samples), compresses models to 3–4 bits per weight, and completes in minutes even for 175 B-parameter networks. On LLaMA-family models, 4-bit GPTQ achieves perplexity within 0.5 points of the uncompressed baseline on WikiText-2, while reducing the weight memory footprint by approximately 4×.

AWQ. Lin *et al.* [83] argue that not all weight channels contribute equally to model output: a small fraction of channels corresponding to large-magnitude activations are disproportionately salient. Activation-Aware Weight Quantization (AWQ) identifies these salient channels via a calibration pass and applies per-channel scaling factors that effectively *protect* salient weights from quantization error by expanding their dynamic range prior to quantization. Because the scaling is absorbed into adjacent layers, no additional computation is required at inference time. AWQ achieves strong accuracy retention at both 4-bit and 3-bit precision across instruction-tuned and chat models, and its weight-only, channel-wise granularity maps cleanly onto NPU weight-stationary datapaths.

4.1.2. Quantization-Aware Training (QAT)

While PTQ is convenient, QAT incorporates quantization noise into the training loop, allowing the model to adapt its weight distribution to the target precision. This yields higher accuracy at extreme bit-widths but requires access to training data and compute.

BitNet. Wang *et al.* [122] propose training Transformers *from scratch* with binarized (± 1) weights. BitNet replaces standard linear projections with *BitLinear* layers that binarize weights using a sign function during the forward pass and maintain latent full-precision weights for gradient updates via straight-through estimators. By constraining weights to a single bit, all weight multiplications reduce to additions and subtractions, eliminating the need for multiply–accumulate (MAC) units entirely. Despite the extreme compression, BitNet models trained at scale exhibit surprisingly competitive perplexity, particularly when accompanied by 8-bit activation quantization to stabilize training.

BitNet b1.58. Ma *et al.* [93] extend the BitNet paradigm from binary to *ternary* weights constrained to $\{-1, 0, +1\}$, a scheme they denote as 1.58-bit (since $\log_2 3 \approx 1.58$). The addition of zero as a weight value introduces an implicit form of structured sparsity—zero-valued weights skip computation entirely—while retaining the addition-only arithmetic benefit. Trained from scratch at the 3 B-parameter scale, BitNet b1.58 matches the perplexity of full-precision LLaMA models of comparable size while offering theoretical energy savings exceeding 70× for matrix multiplications on custom hardware. The ternary weight format is also highly amenable to NPU implementation, as it requires only trivial look-up operations rather than general-purpose multipliers.

4.1.3. Activation-Aware Methods

SmoothQuant. Xiao *et al.* [131] identify a fundamental asymmetry in Transformer quantization: weights are relatively easy to quantize, but activations contain per-channel outliers that resist low-bit representation. SmoothQuant resolves this by applying a *mathematically equivalent transformation* that migrates the quantization difficulty from activations to weights. Concretely, for a linear layer $\mathbf{Y} = \mathbf{X}\mathbf{W}$, the method introduces per-channel scaling factors $\mathbf{s}$ such that $\mathbf{Y} = (\mathbf{X}\,\mathrm{diag}(\mathbf{s})^{-1})(\mathrm{diag}(\mathbf{s})\,\mathbf{W})$. The scaled activations $\hat{\mathbf{X}}$ become smoother (reduced outlier magnitude), while the scaled weights $\hat{\mathbf{W}}$ absorb the difficulty but remain amenable to quantization due to their naturally lower variance. The scaling factors are determined offline from a small calibration set and fused into preceding layer norms at deployment, incurring zero runtime overhead. SmoothQuant enables W8A8 (8-bit weights, 8-bit activations) quantization of OPT-175B and BLOOM-176B models with less than 1% accuracy loss, unlocking fully integer inference paths that are essential for NPU acceleration.

4.1.4. Mixed-Precision and NPU-Aware Quantization

Uniform bit-width assignment across all layers is suboptimal: early layers, attention projections, and output heads often exhibit different sensitivity to quantization noise. Mixed-precision methods assign heterogeneous bit-widths to maximize accuracy under a given model-size or latency budget.

FlexiQ. FlexiQ [70] proposes an *adaptive per-layer bit-width allocation* strategy guided by the value range and gradient sensitivity of each layer. By profiling the dynamic range of weights and activations on a calibration set, FlexiQ assigns higher precision (8-bit) to sensitive layers—typically the first and last Transformer blocks—and lower precision (4-bit or 2-bit) to the more redundant intermediate layers. The resulting mixed-precision configuration is jointly optimized to meet a target latency constraint on a specific NPU hardware profile, making FlexiQ one of the first methods to treat the quantization policy as a hardware-aware optimization problem rather than a purely statistical one. Evaluated on both vision and language workloads deployed to mobile NPUs, FlexiQ achieves accuracy–latency Pareto fronts that consistently dominate uniform quantization baselines, reducing latency by 15–30% at iso-accuracy or improving accuracy by 1–3 percentage points at iso-latency.

MobileQuant. Tan *et al.* [115] observe that existing PTQ methods often rely on floating-point de-quantization or mixed-type operations that are poorly supported by mobile NPUs, which are optimized for *integer-only* arithmetic. MobileQuant introduces an *equivalent transformation* framework that absorbs all floating-point scaling and zero-point adjustments into integer-friendly operations, enabling fully integer quantized execution on commodity mobile NPUs (Qualcomm Hexagon, MediaTek APU, Samsung Eden). The method co-optimizes weight, activation, and KV-cache quantization parameters to minimize round-trip quantization error while ensuring every operation in the inference graph maps to an NPU-supported integer kernel. On Qualcomm's Snapdragon 8 Gen 3, MobileQuant achieves up to 2.4× speedup over its floating-point baseline with negligible perplexity degradation on LLaMA-7B.

4.1.5. Quantization Methods Summary

Table 3 provides a comparative overview of the quantization methods discussed above. Several observations emerge. First, PTQ methods (*e.g.*, GPTQ, AWQ) offer the most practical deployment path for on-device scenarios since they require no retraining. Second, QAT methods (BitNet, BitNet b1.58) achieve the most aggressive compression but demand full training runs and, at present, lack broad hardware support. Third, NPU compatibility requires careful attention to the arithmetic format: integer-only methods (SmoothQuant, MobileQuant) are directly deployable on current mobile NPUs, while sub-4-bit and ternary schemes depend on specialized hardware or software emulation. Fig. 6 condenses these trade-offs into a decision tree for choosing a quantization strategy under typical deployment constraints and NPU support.

4.2. Pruning and Sparsity

Pruning removes redundant parameters from a trained network to reduce both storage and computation. Two broad

Table 3
Comparison of Quantization Methods for Language Models

Method	Type	Bit-width	Accuracy Retention	Speedup	NPU Compatible
LLM.int8() [27]	PTQ	W8 (mixed FP16)	>99% (perplexity)	~1.2×	Partial (requires FP16 fallback)
GPTQ [37]	PTQ	W4 / W3	>98% at W4	~2–3× (memory)	Yes (weight-only INT4)
AWQ [83]	PTQ	W4 / W3	>98% at W4	~2–3× (memory)	Yes (weight-only INT4)
SmoothQuant [131]	PTQ	W8A8	>99%	~1.5×	Yes (INT8 fully supported)
BitNet [122]	QAT	W1A8	~90–95%	Theoretical >10×	No (custom hardware needed)
BitNet b1.58 [93]	QAT	W1.58A8	~95–98%	Theoretical >10×	No (ternary logic needed)
FlexiQ [70]	PTQ	Mixed (W2–W8)	>97%	Hardware-dependent	Yes (NPU-aware allocation)
MobileQuant [115]	PTQ	W4A8 / W8A8 (int-only)	>98%	~2.4×	Yes (integer-only design)

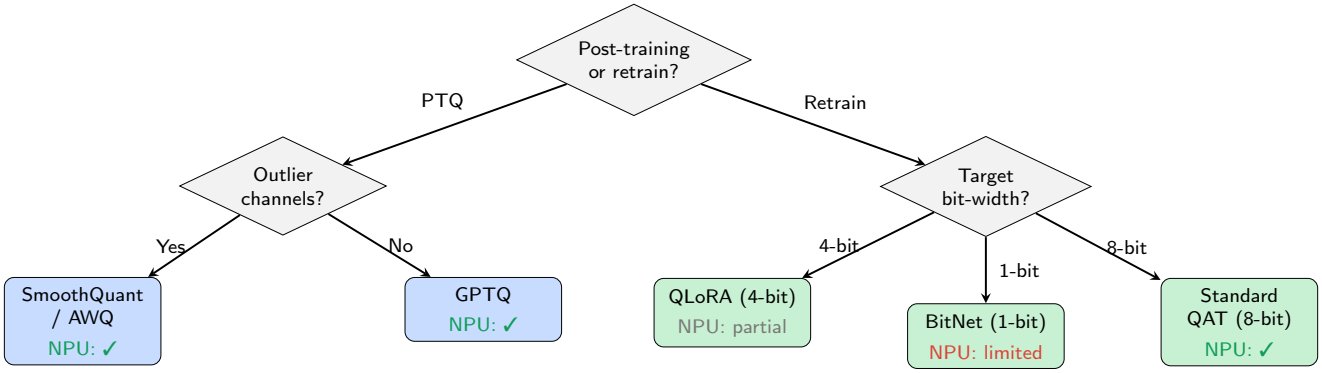


Figure 6: Decision tree for selecting a quantization strategy based on constraints and NPU compatibility.

paradigms exist: *unstructured pruning*, which zeros out individual weights irrespective of their position, and *structured pruning*, which removes entire neurons, attention heads, or layers to produce architecturally smaller models.

4.2.1. Unstructured vs. Structured Pruning

Unstructured pruning typically achieves higher sparsity at a given accuracy level because it is free to remove any weight. However, the resulting irregular sparsity pattern does not map efficiently onto dense matrix hardware: sparse matrix storage formats introduce indexing overhead, and the irregular memory access patterns defeat the cache hierarchies and systolic arrays in modern NPUs. Structured pruning, by contrast, removes contiguous blocks—entire columns of weight matrices, full attention heads, or even complete Transformer layers—yielding smaller dense models that can exploit standard dense kernels without specialized sparse hardware. The trade-off is that structured pruning is more constrained and typically requires more aggressive fine-tuning to recover accuracy lost from removing entire functional units.

4.2.2. One-Shot Pruning at Scale

SparseGPT. Frantar and Alistarh [36] demonstrate that billion-parameter language models can be pruned to 50–60% unstructured sparsity in a single forward pass, without any weight retraining. The key algorithmic insight is to formulate pruning—like quantization in GPTQ—as a layer-wise sparse reconstruction problem and solve it using an approximate second-order method. Specifically, SparseGPT prunes weights column-by-column, at each step selecting the weight with the smallest reconstruction cost to set to zero, then updating the remaining unpruned weights to compensate via the inverse Hessian. On OPT-175B, SparseGPT achieves 50% sparsity with less than 0.5 perplexity increase and 60% sparsity with approximately 1 point of perplexity loss—a remarkable result given that no gradient computation or fine-tuning is performed. The method also composes with quantization: a jointly 50%-sparse, 4-bit model achieves a combined compression ratio exceeding 10× with modest accuracy degradation.

4.2.3. Hardware-Compatible Sparsity Patterns

For sparsity to translate into actual latency and energy savings on NPU hardware, the pruning pattern must align with the accelerator's execution model. The *N:M structured*

sparsity pattern, in which exactly N out of every M consecutive weights are zero, has emerged as the dominant hardware-friendly sparsity format. NVIDIA's Ampere architecture natively supports 2:4 sparsity (50% structured sparsity) with dedicated sparse Tensor Core instructions that provide a 2× throughput improvement over the dense baseline. Qualcomm's Hexagon NPU and ARM's Ethos-U NPUs similarly benefit from block-sparse patterns that align with their fixed-size MAC arrays.

SparseGPT can be configured to produce N:M patterns, though the constrained sparsity structure typically reduces the achievable sparsity level compared to unconstrained pruning at the same accuracy. Recent work explores hybrid strategies that apply N:M sparsity to the most computation-heavy feed-forward layers while leaving attention layers dense, striking a balance between hardware utilization and accuracy preservation.

4.2.4. Layer-Level Pruning

An alternative to weight-level and block-level pruning is *layer-level pruning*, which removes entire layers to produce architecturally shallower models. Lu *et al.* [92] develop a generic layer-pruning method for signal-modulation recognition networks, illustrating how similarity-aware selection can remove redundant depth in compact edge-oriented models. Although that application is not an LLM, it provides a useful structured-pruning precedent. More directly, recent work [91] at ICLR 2026 reassesses layer pruning specifically for LLMs, finding that the sensitivity of individual layers varies dramatically with model scale and training data, and that careful layer selection can remove 20–30% of layers with less than 2% perplexity degradation on models above 7B parameters. For on-device deployment, layer pruning is particularly attractive because it produces standard dense models that require no specialized sparse hardware support—the pruned model runs on existing NPU datapaths with proportionally reduced latency and energy consumption.

4.2.5. Accuracy–Sparsity–Hardware Trade-offs

The practical utility of pruning for on-device deployment is governed by a three-way trade-off. (i) *Accuracy vs. sparsity*: moderate sparsity (40–50%) is largely free for large models, but accuracy degrades sharply beyond 60–70% without task-specific fine-tuning. (ii) *Sparsity vs. hardware support*: only structured and N:M sparsity patterns yield real speedups on current NPU silicon; unstructured sparsity remains largely a storage optimization. (iii) *Composability with quantization*: pruning and quantization are complementary—SparseGPT + GPTQ achieves >10× compression—but the interaction effects on accuracy are non-trivial, and the combined sparse-quantized formats impose additional complexity on the inference runtime.

4.3. Knowledge Distillation and Low-Rank Adaptation

Whereas quantization and pruning compress an existing model, knowledge distillation and low-rank adaptation either *transfer* knowledge to a smaller architecture or *augment* a compressed model with lightweight task-specific parameters.

4.3.1. Low-Rank Adaptation (LoRA)

Hu *et al.* [52] observe that the weight updates during fine-tuning of large language models exhibit low intrinsic dimensionality. LoRA exploits this by freezing the pre-trained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d\times k}$ and injecting a trainable low-rank decomposition $\Delta\mathbf{W} = \mathbf{BA}$, where $\mathbf{B} \in \mathbb{R}^{d\times r}$ and $\mathbf{A} \in \mathbb{R}^{r\times k}$ with rank $r \ll \min(d, k)$. At inference time, the adapter weights are merged into the base weights ($\mathbf{W} = \mathbf{W}_0 + \mathbf{BA}$), incurring zero additional latency or memory overhead. With $r = 8$–64, LoRA reduces the number of trainable parameters by >10,000× compared to full fine-tuning while matching or exceeding its accuracy on downstream tasks.

For on-device deployment, LoRA offers two critical benefits. First, it enables *task-specific customization* of a compressed base model without full retraining—a user or device manufacturer can adapt a quantized SLM to a domain (*e.g.*, medical terminology, smart-home commands) by training only the low-rank adapters. Second, multiple LoRA adapters can be stored and swapped dynamically, enabling a single base model to serve multiple tasks with minimal memory overhead.

4.3.2. QLoRA

Dettmers *et al.* [28] combine aggressive base-model quantization with LoRA fine-tuning to push the memory frontier further. QLoRA quantizes the pre-trained base model to 4-bit precision using a novel *NormalFloat* (NF4) data type that is information-theoretically optimal for normally distributed weights, and then attaches LoRA adapters in BFloat16 precision. A *double quantization* scheme further compresses the quantization constants themselves. Together, these innovations reduce the memory required to fine-tune a 65 B-parameter model to under 48 GB, enabling fine-tuning on a single consumer GPU. For on-device contexts, QLoRA is significant because it demonstrates that high-quality task adaptation can be performed on the same quantized model that will be deployed at inference time, eliminating the train–quantize–deploy pipeline mismatch.

4.3.3. Knowledge Distillation

Knowledge distillation trains a compact *student* model to mimic the output distribution or intermediate representations of a larger *teacher* model. In the context of on-device SLMs, distillation serves two complementary roles. First, it is the primary mechanism by which the performance gap between billion-parameter teachers and sub-billion-parameter students is closed: models such as DistilBERT [105] and MobileBERT [114] retain 95–97% of their teacher's accuracy at 40–60% of the parameter count. Second, distillation can be combined with quantization and pruning in a multi-stage compression pipeline, where the teacher guides the student through both architectural reduction and precision

lowering, mitigating the compounding accuracy losses that arise when the two techniques are applied independently.

Recent work extends distillation to the RAG setting: a cloud-hosted large model can serve as the teacher for an on-device student, distilling not only language modeling capability but also retrieval-grounded reasoning patterns. This teacher–student paradigm aligns naturally with edge–cloud hybrid architectures (discussed in Section 7), where the cloud teacher periodically updates on-device student adapters via over-the-air LoRA patches. MemLoRA [6] instantiates exactly this idea for on-device memory systems, distilling expert LoRA adapters so that a compact student retains specialized capability without hosting the full teacher locally.

Recent reasoning-model studies refine what should be distilled. DRP combines skill-aware step decomposition, inference-time reasoning pruning, and student tuning, cutting GSM8K reasoning from 917 to 328 tokens while improving accuracy and reducing AIME token use by 43% without a performance drop [61]. PerSyn instead routes each prompt to the teacher whose output best balances response quality and student learnability, avoiding the cost of generating every prompt with every candidate teacher [138]. Token-level analysis of on-policy distillation further finds that persistent high-loss "Rock Tokens" can consume substantial optimization bandwidth while contributing little to reasoning performance, and that bypassing them can streamline alignment [62]. Together, these results suggest that sustainable on-device distillation should select the teacher, reasoning steps, and token-level targets jointly rather than treating all teacher outputs as equally useful.

Complementary to model-level distillation, *dataset distillation* [90] offers an orthogonal approach to training efficiency: by synthesizing a compact surrogate dataset that captures the essential information of a much larger corpus, dataset distillation reduces the data requirements for fine-tuning and adaptation. When combined with LoRA or QLoRA, dataset distillation can dramatically reduce the compute required for domain-specific on-device model customization—an important consideration given the sustainability objectives motivating this survey. Furthermore, synergized approaches to data efficiency and compression [75] demonstrate that jointly optimizing data selection and model compression yields better accuracy–efficiency trade-offs than applying either technique in isolation, suggesting that holistic co-optimization of data, model architecture, and compression is the preferred path for sustainable on-device deployment.

4.3.4. Relevance to On-Device Deployment

The combination of LoRA and QLoRA is particularly consequential for the on-device RAG systems reviewed in Section 5. A practitioner can (i) select a pre-trained SLM, (ii) apply GPTQ or AWQ to reduce it to 4-bit weights, (iii) attach LoRA adapters for domain-specific fine-tuning, and (iv) deploy the merged model on an NPU. This pipeline preserves the base model's general capability while adding domain specialization at negligible cost, and the entire fine-tuning step requires only modest compute thanks to QLoRA's memory efficiency.

4.4. Inference Optimization Techniques

Beyond static model compression, a family of runtime optimization techniques addresses the computational and memory bottlenecks that arise during autoregressive inference on constrained hardware.

4.4.1. KV Cache Management

Autoregressive Transformer decoding is fundamentally *memory-bound*: at each token generation step, the model must read the entire key–value (KV) cache accumulated from all previous tokens and the full model weights, but performs relatively little arithmetic per byte read. For a model with L layers, hidden dimension d, and context length n, the KV cache occupies $2Lnd$ elements per precision byte, which for a 7 B-parameter model at 2,048-token context in FP16 exceeds 2 GB—a significant fraction of the total memory on a mobile device. Effective KV cache management strategies include: (i) *quantized KV caches* (INT8 or INT4) to halve or quarter the memory footprint, as exploited by MobileQuant [115]; (ii) *sliding-window attention* that bounds cache size to a fixed window, discarding the oldest entries; and (iii) *paged attention* schemes that manage cache memory in fixed-size blocks to eliminate fragmentation.

4.4.2. Flash Attention

The standard self-attention computation requires materializing the $n \times n$ attention matrix in GPU high-bandwidth memory (HBM), resulting in $\mathcal{O}(n^2)$ memory consumption and excessive HBM reads/writes that dominate wall-clock time.

FlashAttention. Dao *et al.* [26] propose an *IO-aware* exact attention algorithm that computes attention in a tiled, fused fashion. By partitioning the queries, keys, and values into blocks that fit in on-chip SRAM and applying the online softmax trick to accumulate the softmax statistics incrementally, FlashAttention avoids ever materializing the full $n \times n$ matrix in HBM. The result is a 2–4× wall-clock speedup over standard attention on GPUs and, critically, a reduction in HBM accesses from $\mathcal{O}(n^2)$ to $\mathcal{O}(n^2 d / M)$, where M is the SRAM size. This IO reduction translates directly into energy savings, as DRAM accesses dominate the energy budget of modern accelerators.

FlashAttention-2. Dao [25] refines the algorithm with improved work partitioning across GPU thread blocks and reduced non-matmul floating-point operations, achieving an additional 2× speedup over the original FlashAttention and reaching 50–73% of theoretical peak FLOPS on A100 GPUs. For on-device NPUs, the core tiling principle of FlashAttention—keeping data in fast on-chip memory and minimizing off-chip transfers—is directly applicable, though the specific tiling dimensions must be adapted to the NPU's SRAM geometry and DMA engine capabilities.

4.4.3. Flash-Memory-Aware Inference

LLM in a Flash. Alizadeh *et al.* [2] address a distinct bottleneck in on-device inference: loading model weights from secondary storage (flash memory/SSD) into DRAM when the model exceeds available RAM. They propose two complementary strategies: (i) *windowed loading*, which keeps only the active subset of layers in DRAM and pages the rest from flash, and (ii) *row–column bundling*, which restructures the sparse weight matrix to co-locate non-zero weights in flash-friendly contiguous reads, dramatically improving flash read throughput. When combined with sparsity predictor networks that identify which weight rows will be activated before loading, the method enables inference of models up to 2× the available DRAM capacity with only modest latency overhead. This is particularly relevant for on-device scenarios where smartphones typically have 6–12 GB of DRAM shared across all running applications.

4.4.4. Speculative Decoding

Speculative decoding addresses the latency problem of autoregressive generation from a different angle: rather than accelerating each token's generation step, it *reduces the number of serial steps*. A small, fast *draft* model generates a block of k candidate tokens autoregressively, and the larger *target* model then verifies all k tokens in a single forward pass. Tokens that agree with the target model's distribution are accepted; the first disagreement triggers a resampling step from the target model's corrected distribution. Because verification is parallelizable (all k tokens are processed simultaneously via a single forward pass), the effective tokens-per-second throughput increases by a factor proportional to the draft model's acceptance rate. In on-device settings, a tiny 100 M-parameter draft model paired with a 3 B-parameter target model can achieve 2–3× latency reduction with identical output quality, a particularly valuable trade-off when the target model is memory-bound.

4.4.5. Operator Fusion and Graph Optimization

Mobile inference runtimes—TensorFlow Lite, ONNX Runtime Mobile, ExecuTorch, QNN (Qualcomm Neural Network SDK)—apply graph-level optimizations that are essential for extracting peak performance from NPU hardware. *Operator fusion* merges sequences of element-wise, normalization, and activation operations into a single NPU kernel launch, eliminating intermediate memory round-trips. For Transformer models, common fusion patterns include fusing layer normalization with subsequent linear projections, fusing the query–key–value projections into a single batched matrix multiply, and fusing the softmax and value multiplication within the attention block. *Graph optimization* passes further rewrite the computation graph to exploit NPU-specific instructions: converting dynamic shapes to static padded shapes for predictable tiling, replacing general matrix multiplications with NPU-native convolution instructions when dimensions align, and inserting explicit quantize/dequantize nodes that map onto the NPU's fixed-point data paths. Together, these runtime optimizations typically yield an additional 1.5–3× speedup on top of the gains from model-level compression techniques.

4.5. Summary

The compression and optimization techniques reviewed in this section form a composable toolkit for on-device model deployment. In practice, state-of-the-art on-device pipelines stack multiple techniques: a 4-bit GPTQ- or AWQ-quantized base model with LoRA domain adapters, 50% N:M structured sparsity from SparseGPT, FlashAttention-derived tiled attention, quantized KV caches, and fused operator graphs targeting the device's NPU. The cumulative effect can compress a 7 B-parameter model from approximately 14 GB (FP16) to under 3 GB while maintaining over 95% of baseline accuracy and reducing per-token energy consumption by an order of magnitude. These compressed models form the generative backbone of the on-device RAG systems examined in the following section.

5. On-Device RAG Systems

Fig. 7 contrasts (a) cloud-based and (b) on-device RAG pipelines, highlighting how edge deployment removes wide-area network dependence and can improve privacy and per-query energy, at the cost of tight on-device memory and compute budgets. Moving RAG pipelines from data-center infrastructure to edge devices fundamentally reshapes every stage of the retrieve-then-generate paradigm. Cloud-based RAG enjoys virtually unbounded GPU memory, high-throughput vector databases, and large reranking models; on-device RAG must deliver comparable answer quality within a fixed memory envelope (typically 4–8 GB shared among the operating system, the language model, its KV cache, the embedding model, and the vector index), under strict latency and thermal budgets, and frequently without network connectivity. This section examines the architectural adaptations (Section 5.1), complete system implementations (Section 5.2), efficiency-oriented algorithmic innovations (Section 5.3), and outstanding failure modes (Section 5.4) that collectively define the state of the art in on-device RAG. The systems surveyed here correspond primarily to taxonomy category F (On-Device RAG, 19 papers), building on the RAG-core work of category E (20 papers); the full per-paper classification appears in Table 13 (Appendix).

5.1. RAG Pipeline Adaptations for Edge

A standard cloud RAG pipeline comprises four stages: *indexing* (chunking documents and computing dense embeddings), *retrieval* (approximate nearest-neighbor search over a vector store), *optional reranking* (cross-encoder scoring of top-k candidates), and *generation* (feeding retrieved context to a large language model). Each stage imposes assumptions that break on edge hardware.

Lightweight embedding models. Cloud deployments commonly employ large bi-encoder models with 300M–1B parameters (*e.g.*, E5-large, GTE-Qwen2) to produce

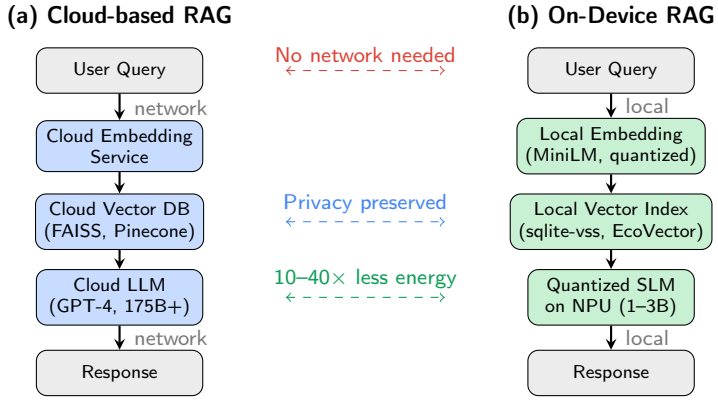


Figure 7: Comparison of (a) cloud-based RAG pipeline and (b) on-device RAG pipeline. On-device deployment eliminates network latency and data transmission, preserves privacy, and leverages NPU acceleration for energy-efficient inference.

high-quality dense representations. On-device settings require bi-encoders in the 20–80M parameter range, such as MiniLM-L6 [123], all-MiniLM-L6-v2, or quantized variants of BGE-small. MobileRAG [99] further proposes training-free *embedding model adaptation* via knowledge distillation from a large teacher encoder, narrowing the recall gap to within 2% of the teacher while fitting within 50 MB of storage. Quantizing embedding weights to INT8 yields an additional 2–4× memory reduction with negligible impact on retrieval recall [83]. Importantly, the choice of embedding paradigm itself warrants careful evaluation for on-device use cases—the MTEB benchmark [95] shows that no single embedding model dominates across retrieval, clustering, and classification tasks, so the smallest model adequate for the target task, rather than the highest-ranked overall, is the appropriate on-device choice. Consistent with this, empirical evidence on short-text industrial corpora from our own prior work [72] demonstrates that lightweight static embeddings (Word2Vec) can outperform Transformer-based sentence encoders (Sentence-BERT) by over 350% on clustering tasks when texts are short and domain-specific, suggesting that on-device RAG systems operating over concise documents may benefit from simpler, more resource-efficient embedding strategies rather than defaulting to the largest available Transformer encoder.

On-device vector search. Full-scale FAISS [64] assumes GPU-backed flat or IVF indices and memory-mapped storage, a deployment profile that smartphones cannot match. Several lightweight alternatives have emerged:

- **FAISS-lite** strips the library to CPU-only flat and IVF-PQ indices, enabling sub-second search over corpora at the 10^5 scale on ARM processors.
- **sqlite-vss** integrates vector similarity search directly into SQLite, leveraging the database engine already present on every mobile OS and avoiding a second runtime dependency.
- **EcoVector** [99] introduces a cache-friendly *neighbor-aware vector pruning* strategy that reduces index size by up to 70% while maintaining 95% recall@10 on standard benchmarks, specifically designed for memory-constrained mobile deployment.

Context compression and selective retrieval. Even after retrieval, the volume of context forwarded to the language model must be tightly controlled. Long contexts inflate KV cache memory, slow autoregressive decoding, and—as shown by Liu *et al.* [85]—degrade answer quality when salient information is buried in the middle of the context window. Strategies include *extractive compression* (selecting only the most relevant sentences or paragraphs from retrieved passages), *abstractive compression* (using a lightweight summarizer to distill passages before prompting), and *token-level pruning* (dropping low-attention tokens during prefill).

AttentionRAG [34] introduces an *attention-guided context pruning* mechanism that leverages the generator's own attention patterns to identify and remove passages that receive negligible attention during a lightweight preliminary forward pass. By pruning context at the passage level before the full generation step, AttentionRAG reduces context length by 30–50% while maintaining answer faithfulness—a technique directly applicable to on-device settings where prefill compute dominates energy consumption. SWE-Pruner [126] extends context pruning to agentic settings, proposing self-adaptive pruning strategies for coding agents that dynamically adjust the aggressiveness of context removal based on task complexity. While designed for software engineering tasks, the underlying principle—that context pruning intensity should adapt to query difficulty—generalizes to on-device RAG, where simple factoid queries can tolerate aggressive pruning while complex multi-hop questions require richer context. MobileRAG [99] formalizes a related approach as *Selective Content Reduction* (SCR), retaining only the top-scoring sentence clusters within each passage and reducing the average context length by 40–55% with less than 1% accuracy degradation.

Memory budget partitioning. On a device with 6 GB of available RAM, a typical allocation might be: quantized SLM weights (1.5–2.5 GB for a 3B model at 4-bit), KV

cache (0.3–1.0 GB depending on context length and batch size), embedding model (50–200 MB), and vector index (100 MB–1 GB depending on corpus size). These four components compete for a shared memory pool, making joint optimization essential. EdgeRAG [107] and PocketRAG [67] both adopt *lazy loading* strategies that keep only a subset of index shards resident, paging in additional clusters on demand to smooth peak memory usage.

5.2. End-to-End On-Device RAG Frameworks

Several recent systems have demonstrated that end-to-end RAG is feasible on consumer-grade mobile and edge hardware. We review four representative frameworks in detail and synthesize their design choices in Table 4.

MiniRAG. Fan *et al.* [33] observe that conventional RAG pipelines rely heavily on the semantic understanding capabilities of large LMs during both indexing and retrieval, making them brittle when paired with small language models. MiniRAG addresses this through two innovations: (i) a *semantic-aware heterogeneous graph* that indexes text chunks, named entities, and their co-occurrence relations as a unified graph structure, enabling topology-guided retrieval that compensates for weaker embedding quality; and (ii) a *lightweight semantic-aware query routing* mechanism that adaptively chooses between graph-based and embedding-based retrieval paths depending on query complexity. On five benchmark datasets, MiniRAG achieves accuracy comparable to cloud-scale RAG while requiring only 25% of the storage footprint of conventional indexing baselines, making it well-suited to storage-limited edge devices.

EdgeRAG. Seemakhupt *et al.* [107] target the latency bottleneck of loading large HNSW or IVF indices from flash storage. EdgeRAG introduces *intra-cluster embedding pruning*, which identifies and discards redundant embeddings within each Voronoi cell of an IVF index, reducing index size and thus I/O latency. For the pruned embeddings, the system performs *on-demand embedding generation* at query time by running the cached document text through the lightweight bi-encoder only when the coarse-grained cluster assignment suggests relevance. An *adaptive cache* retains frequently accessed cluster centroids and hot embeddings in DRAM, amortizing flash-read costs across queries. In experiments on a Qualcomm Snapdragon 8 Gen 2 platform, EdgeRAG reduces first-token latency by 1.35–3.0× compared to naïve IVF retrieval while maintaining equivalent retrieval recall.

PocketRAG. PocketRAG [67] is an Android-native offline RAG application targeting first-aid and emergency medical guidance—a domain where network connectivity cannot be assumed. The system packages a quantized Phi-3-mini (4-bit, 2.2 GB) as the generator, an INT8 MiniLM embedding model, and a sqlite-vss vector index of 15,000 first-aid passages, all within a 3.4 GB APK. PocketRAG introduces a *hybrid retrieval* strategy combining BM25 keyword matching with dense semantic search and a lightweight cross-attention reranker distilled to 12M parameters. User studies with 120 participants demonstrate 94.5% factual accuracy on a curated first-aid QA benchmark, a 4× end-to-end latency speedup over cloud-based GPT-4 RAG (measured under typical 4G connectivity), and full offline functionality. The work highlights the viability of on-device RAG for safety-critical applications in connectivity-constrained environments.

MobileRAG. MobileRAG [99] presents a holistic optimization stack spanning every pipeline stage. At the indexing layer, *EcoVector* reduces vector index memory through neighbor-aware pruning. At the retrieval layer, a cache-aware search policy batches I/O operations to minimize random flash reads. At the generation layer, *Selective Content Reduction* compresses retrieved context before it enters the SLM, lowering both prefill compute and KV cache pressure. Evaluated on a Samsung Galaxy S24 (Snapdragon 8 Gen 3, 8 GB RAM) with a 4-bit Gemma-2B generator, MobileRAG achieves 8.89× lower end-to-end latency than an unoptimized baseline and 24–40% reduction in per-query power consumption, while matching the baseline's answer accuracy within a 1.2% margin. The power savings translate directly to extended battery life—a metric of increasing importance as on-device AI workloads grow.

Cross-cutting observations. Several design principles emerge across these systems. First, all four eschew heavyweight reranking stages in favor of retrieval-integrated relevance signals (graph topology in MiniRAG, cluster-level pruning in EdgeRAG, lightweight cross-attention in PocketRAG, and content-level scoring in MobileRAG). Second, every system adopts aggressive quantization of both the generator SLM and the embedding model, confirming that 4-bit weight quantization is now the de facto standard for on-device deployment (see Section 4). Third, offline capability is universal, underscoring the importance of network-independent operation as a first-class design requirement rather than a fallback mode.

5.3. RAG Efficiency Techniques

Beyond integrated systems, several algorithmic innovations target specific bottlenecks in the RAG pipeline, many of which are directly applicable to on-device settings.

Sparse context selection. SparseRAG [146], presented at ICLR 2025, reconceptualizes the context utilization problem in RAG. Standard RAG concatenates all retrieved passages into a single context window, forcing the LM to attend to every token even when only a fraction is relevant. SparseRAG introduces *parallel context encoding with block-wise sparsity masks*, where each retrieved passage is encoded independently with full self-attention but cross-passage attention is zeroed out. A set of learned *control tokens* (analogous to special tokens) is prepended to each block; during a *selective decoding* phase, the model attends only to blocks whose control tokens signal high relevance. This yields two benefits: (i) irrelevant passages are effectively pruned from the decoding computation, reducing FLOPs proportionally to the fraction of discarded blocks; and (ii) the model is less

Table 4
Comparison of On-Device RAG Systems

System	SLM Used	Retrieval Method	Vector DB	Latency	Memory	Accuracy	Offline
MiniRAG [33]	Qwen2-0.5B / Phi-3-mini	Heterogeneous graph + semantic routing	Custom graph index	~2 s / query	~3.2 GB	Comparable to cloud baselines	✓
EdgeRAG [107]	Phi-3-mini (4-bit)	Pruned IVF + on-demand embedding	FAISS IVF-PQ (pruned)	1.35–3× faster	~4.0 GB	Equivalent recall to baseline	✓
PocketRAG [67]	Phi-3-mini (4-bit)	BM25 + dense + cross-attn reranker	sqlite-vss	4× faster than cloud	3.4 GB (APK)	94.5% factual accuracy	✓
MobileRAG [99]	Gemma-2B (4-bit)	EcoVector + cache-aware search	EcoVector index	8.89× lower	~3.8 GB	–1.2% vs. baseline	✓

susceptible to the "lost in the middle" phenomenon [85] because each passage is encoded in isolation. On Natural Questions and TriviaQA, SparseRAG improves EM scores by 3–5% over dense concatenation while reducing generation-phase FLOPs by 35–50%.

Speculative RAG. Speculative RAG [128], also at ICLR 2025, transfers the speculative decoding paradigm from token-level generation to document-level RAG verification. A smaller *specialist drafter* model (1–3B parameters) independently generates candidate answers from distinct subsets of the retrieved documents, producing multiple draft responses in parallel. A larger *generalist verifier* model then scores each draft for factual consistency and selects the best response. Because the drafter is lightweight and runs in parallel across document subsets, the overall wall-clock time is dominated by the single verifier pass rather than by sequential generation over the full context. In evaluation, Speculative RAG achieves a 12.97% accuracy improvement over standard RAG on knowledge-intensive benchmarks and a 50.83% reduction in end-to-end latency. For on-device deployment, the drafter can be mapped to the device's SLM while the verifier runs on a more capable edge server or cloud endpoint, enabling a tiered hybrid architecture.

Multi-hop retrieval for complex queries. Multi-hop questions pose a particular challenge for on-device RAG because they require evidence synthesis across multiple passages or reasoning chains, while limited retrieval depth and shorter context windows constrain the system's ability to gather and integrate dispersed evidence. Reasoning in Trees [110] addresses this through a tree-structured retrieval approach that decomposes complex queries into sub-questions, retrieves evidence for each sub-question independently, and aggregates the results through a structured reasoning process. This decompose-retrieve-aggregate paradigm is well-suited to on-device deployment because sub-queries can be processed with smaller context windows, reducing peak memory and energy consumption per retrieval step while improving accuracy on multi-hop benchmarks.

Hybrid retrieval and document routing. Beyond the dense-sparse retrieval dichotomy, recent work explores hybrid architectures that combine document-level routing with chunk-level retrieval to address failure modes specific to structurally homogeneous corpora. Cheng *et al.* [20] (authors' prior work) identify a robustness–precision trade-off in financial RAG: chunk-based retrieval achieves high precision but suffers from cross-document confusion, while document-level routing eliminates confusion but sacrifices targeted precision. Their Hybrid Document-Routed Retrieval (HDRR) architecture resolves this trade-off by using LLM-based document routing as a first-stage filter followed by scoped chunk retrieval within the identified document, achieving 25% higher accuracy than standard chunk-based RAG while maintaining compact context windows. For on-device RAG, scoped retrieval is particularly beneficial: by restricting vector search to a single document's chunks, the retrieval index can be substantially smaller, reducing both memory footprint and search latency. The importance of retrieval pipeline design is further validated by empirical studies demonstrating that neural reranking yields a 15.5 percentage-point improvement in answer correctness for domain-specific RAG [21], though the computational cost of cross-encoder reranking must be carefully managed on resource-constrained devices.

Difficulty-aware query orchestration. Not all queries impose the same computational burden. Su *et al.* [112] propose difficulty-aware agentic orchestration that predicts query complexity and dynamically selects the retrieval and generation strategy accordingly—routing simple queries through a lightweight path and complex queries through a more capable (but more expensive) pipeline. On edge devices, this principle translates directly into energy savings: if 60–70% of user queries are simple enough for single-passage retrieval with a compact SLM, reserving the full multi-stage pipeline for difficult queries can reduce average energy per query by 30–40% without degrading user-perceived quality.

Modular RAG toolkits. FlashRAG [63] provides a modular open-source framework implementing over 23 RAG algorithms under a unified API, including naïve RAG, iterative

retrieval, self-RAG, and corrective RAG. The toolkit standardizes dataset preprocessing, retrieval configuration, and generation evaluation, facilitating reproducible comparison of RAG strategies. While not specifically designed for edge deployment, FlashRAG's modular architecture enables rapid prototyping of lightweight pipeline variants—for instance, swapping a large cross-encoder reranker for a bi-encoder rescorer or replacing a heavy generator with a quantized SLM—making it a valuable experimentation platform for on-device RAG research.

RL-based document selection for edge RAG. SRAS [96] specifically targets the edge RAG setting with a reinforcement-learning-based lightweight document selector. The policy network, occupying only 0.76 MB, learns to select the minimal subset of retrieved passages that maximizes downstream answer quality, formulating document selection as a sequential decision process with a reward signal combining answer accuracy and passage count penalties. On a Raspberry Pi 4, SRAS achieves sub-1 s document selection latency while retaining 95% of the accuracy obtained when forwarding all retrieved passages to the generator. The negligible size of the policy network makes it an attractive "plug-in" module for any on-device RAG pipeline seeking to reduce context length without retraining the base SLM.

5.4. Challenges and Failure Modes

Despite rapid progress, on-device RAG systems inherit the failure modes of cloud RAG while introducing new challenges unique to resource-constrained environments.

Inherited RAG failure taxonomy. Barnett *et al.* [5] identify seven systematic failure points in RAG systems: (FP1) *missing content*—the knowledge base lacks the information needed to answer the query; (FP2) *missed top-ranked documents*—relevant documents exist but are not retrieved in the top-k; (FP3) *context overflow*—the retrieved context exceeds the model's effective context window; (FP4) *extraction failure*—the model fails to locate the answer within the provided context; (FP5) *wrong format*—the answer is correct but presented in an unexpected format; (FP6) *incomplete answers*—the response addresses only part of a multi-faceted query; and (FP7) *incorrect specificity*—the answer is at the wrong level of detail. On-device RAG systems are disproportionately exposed to FP1 (smaller corpora), FP2 (weaker embedding models reduce retrieval recall), and FP3 (shorter effective context windows due to memory constraints), making mitigation of these failure points a priority for edge-specific system design.

Robustness under noisy retrieval. Empirical studies on retrieval-augmented language models [94] demonstrate that the inclusion of irrelevant or contradictory passages can actively *degrade* model accuracy below the no-retrieval baseline—a phenomenon termed *retrieval pollution*. This risk is amplified on-device, where smaller SLMs possess weaker parametric knowledge and are therefore more susceptible to being misled by noisy context. Robust document filtering, either via trained classifiers or the RL-based selection of SRAS [96], is consequently even more critical in on-device settings than in cloud deployments. Beyond content noise, recent work reveals that even the *tone* of retrieved passages can influence LLM outputs [9], with models exhibiting sensitivity to stylistic variation in the prompt—an effect that compounds with retrieval noise when diverse passage styles are concatenated as context.

Knowledge conflict makes this failure mode harder to diagnose because a correct final answer does not reveal whether the generator followed retrieved evidence or relied on its parametric prior. Chen *et al.* [14] introduce a context-compliance diagnostic that decomposes contextual and prior answers and perturbs the conflict-resolution trace, showing that misleading retrieval can sharply degrade standard RAG while superficially similar answers can arise from different mechanisms. Even a relevant citation may be insufficient: Qian *et al.* [101] identify *citation laundering*, in which a topically related source is attached to a claim stronger than the evidence warrants, and propose evidence-force calibration across relation, modality, scope, temporal-validity, and numeric-specificity dimensions. These results motivate conflict-aware filtering and warrant-strength evaluation for on-device RAG, where limited compute makes repeated retrieval and heavyweight verification especially costly.

On-device-specific challenges. Beyond the general RAG failure taxonomy, on-device systems face several additional constraints:

- **Limited retrieval corpus.** On-device corpora are orders of magnitude smaller than cloud knowledge bases, increasing the probability of FP1 (missing content) and limiting the system to narrow-domain applications or requiring periodic synchronization with a cloud-hosted master index.
- **No reranker budget.** Cross-encoder rerankers, which add 50–200 ms per query on a GPU, may take seconds on a mobile CPU, making them impractical. Systems must rely on single-stage retrieval or very lightweight rescoring.
- **Index staleness.** Without continuous network access, the local index can become stale. Incremental index update mechanisms that avoid full re-indexing are needed but remain under-explored.
- **Thermal throttling.** Sustained RAG workloads can trigger thermal throttling on mobile SoCs, causing latency spikes that degrade user experience during extended sessions.
- **Heterogeneous hardware.** The diversity of NPU, GPU, and DSP configurations across device generations creates a fragmented optimization landscape, complicating portable deployment.

The quality–efficiency Pareto frontier. Ultimately, every on-device RAG design decision represents a point on a multi-dimensional Pareto frontier trading off answer quality against latency, memory, energy, and thermal headroom.

Increasing the retrieval corpus improves coverage (reducing FP1) but inflates index memory; adding a reranker improves precision (reducing FP2) but increases latency and energy; extending the context window reduces FP3 but demands more KV cache memory and slows decoding. No single configuration dominates across all dimensions. We argue that making this trade-off space explicit—through systematic benchmarking with metrics such as accuracy, latency, energy per query, and peak memory (the ALEMC framework proposed in Section 7)—is essential for guiding practitioners toward deployments that are not only functional but also *sustainable* in terms of device battery life and environmental carbon cost.

6. Green AI: Energy and Carbon in the AI Lifecycle

The computational demands of modern language models carry a significant and growing environmental cost. Strubell *et al.* [111] brought early attention to this issue by estimating that training a single large Transformer model can emit as much carbon as five automobiles over their lifetimes. More recently, de Vries [120] projected that AI-related electricity consumption could rival that of entire nations by the late 2020s if current scaling trends continue. In response, the Green AI movement—formalized by Schwartz *et al.* [106]—advocates for reporting efficiency alongside accuracy, shifting the research community's incentive structure away from "Red AI" (pursuing state-of-the-art performance at any computational cost) toward models and systems that achieve strong results with minimal resource expenditure.

This section surveys the tools, strategies, and analytical frameworks that enable practitioners to measure, reduce, and reason about the energy and carbon footprint of AI inference—with particular emphasis on techniques relevant to on-device deployment.

6.1. Energy Measurement Frameworks

Accurate measurement is a prerequisite for optimization. A growing ecosystem of open-source tools enables researchers to estimate the energy consumption and carbon emissions of AI workloads at varying levels of granularity.

CO2-Meter [40] is a comprehensive carbon footprint estimator designed specifically for LLM inference on edge devices. Unlike tools that focus solely on GPU power draw, CO2-Meter accounts for both *operational carbon* (energy consumed during inference) and *embodied carbon* (emissions from device manufacturing and disposal). Critically, it decomposes inference into its constituent phases—prefill and decode—and applies phase-specific energy models that reflect the distinct computational profiles of each stage. The prefill phase, which processes the full input context in parallel, is typically compute-bound, whereas the decode phase, which generates tokens autoregressively, is memory-bandwidth-bound. By modeling these phases separately, CO2-Meter provides more accurate estimates than tools that treat inference as a monolithic workload.

EnviroLLM [3] is an open-source toolkit for tracking the carbon footprint of locally deployed language models. It provides real-time monitoring of energy consumption during inference sessions and incorporates regional grid carbon intensity data to convert energy measurements into CO_2-equivalent estimates. EnviroLLM also supports personalized optimization recommendations, suggesting configuration changes (*e.g.*, batch size, quantization level, context length) that reduce emissions for a given user's workload profile and geographic location.

CodeCarbon [23] has become the most widely adopted carbon tracking library in the machine learning community. Implemented as a lightweight Python package, it estimates CO_2 emissions by combining hardware power measurements (via Intel RAPL, NVIDIA SMI, or TDP-based approximations) with location-aware carbon intensity data. CodeCarbon integrates seamlessly into training and inference loops, requiring only a few lines of instrumentation code. Its broad adoption has made it a de facto standard for reporting emissions in ML papers, though its reliance on software-level power estimation introduces measurement uncertainty on the order of 10–20% compared to hardware power meters.

Zeus [135] takes a complementary approach by focusing on GPU energy optimization during DNN training. Rather than passively measuring emissions, Zeus actively reduces energy consumption through dynamic power management—adjusting GPU power limits and batch sizes to minimize energy per training job while respecting throughput constraints. While primarily targeting training workloads, the principles of dynamic power capping generalize to inference serving, particularly on GPU-equipped edge servers.

A common limitation of these tools is that they instrument a *model*, whereas an on-device RAG deployment is a *pipeline*: query embedding, index search, context assembly, prefill, and decode each draw power differently, and the retrieval stages are typically invisible to accelerator-scoped counters. Measuring the assembled pipeline on the mobile silicon it will actually run on is therefore a distinct exercise from measuring generator inference alone; our own prior work reports such a whole-system energy characterization of an on-device RAG stack on a Snapdragon X Elite NPU platform [19], and the empirical study in Section 7.3 extends that measurement methodology across three hardware tiers.

To enable meaningful comparisons across systems and deployment contexts, the community has converged on several standardized metrics:

- **kWh per inference**: Total energy consumed per query or generation session, measured in kilowatt-hours. For on-device SLMs, typical values range from 10^{-5} to 10^{-3} kWh per query depending on model size and context length.
- **gCO_2e per query**: Grams of CO_2-equivalent emissions per inference, computed by multiplying energy consumption by the local grid carbon intensity (gCO_2/kWh). This metric varies dramatically by

Table 5
Comparison of Green AI Measurement Tools

Tool	Metrics Measured	Platform	Granularity	Open Source
CO2-Meter [40]	Operational + embodied carbon, phase energy	Edge devices (CPU, GPU, NPU)	Per-phase (prefill/decode)	Yes
EnviroLLM [3]	Energy (kWh), CO_2e, grid carbon intensity	Local LLM deployments	Per-session, real-time	Yes
CodeCarbon [23]	Energy (kWh), CO_2e, power draw	CPU, GPU (any platform)	Per-function / per-epoch	Yes
Zeus [135]	GPU energy (J), power cap optimization	NVIDIA GPUs	Per-job / per-batch	Yes

geography—from under 20 gCO_2/kWh in Norway (hydroelectric) to over 700 gCO_2/kWh in Poland (coal-heavy).

- **Power Usage Effectiveness (PUE)**: The ratio of total facility energy to IT equipment energy, capturing cooling and infrastructure overhead. Data centers typically report PUE values between 1.1 (hyperscale) and 1.6 (enterprise). On-device inference implicitly achieves a PUE near 1.0, since the device's thermal management is negligible relative to compute energy.
- **FLOPS per watt**: Computational throughput normalized by power consumption, reflecting hardware energy efficiency. Modern NPUs achieve 5–15 TOPS/W at INT8 precision, substantially exceeding GPU efficiency at comparable precision levels.

Table 5 summarizes the key characteristics of these measurement tools.

Despite this progress, significant measurement gaps remain. Most tools rely on software-level power estimation rather than direct hardware measurement, introducing systematic errors that are difficult to characterize. NPU energy consumption is particularly underserved: vendor-specific power monitoring interfaces are often proprietary or absent, forcing researchers to rely on board-level power measurements that conflate NPU, CPU, and memory contributions. Standardizing energy measurement for heterogeneous SoCs—where inference may span CPU, GPU, NPU, and memory subsystems within a single query—remains an open challenge.

6.2. Sustainable Inference Strategies

Measurement alone is insufficient; the Green AI agenda ultimately requires techniques that reduce the energy and carbon cost of inference while preserving acceptable quality. We identify five complementary strategy families.

Retrieval-augmented generation as computational parsimony. Before considering hardware-level or scheduling-level optimizations, the most structural lever for reducing inference energy is the size of the context itself. In purely parametric long-context inference, the model must attend to every token in a prompt that has been padded with potentially tangential background material, yielding prefill cost that grows linearly—and for standard attention quadratically—with context length, and KV cache energy that accumulates across every decode step. RAG reframes this problem: rather than feeding the model an indiscriminately long prompt, a lightweight retriever selects only the passages most relevant to the current query, producing a compact, targeted context. The resulting reduction in prefill tokens and KV cache footprint translates directly into lower energy per query, so retrieval acts as a *computational parsimony* mechanism in addition to a grounding mechanism. This sustainability advantage is reinforced by the "lost in the middle" phenomenon [85]: dumping a long context into the model often fails to even exploit the additional tokens, meaning that long-context prompting can simultaneously burn more energy *and* degrade answer quality relative to a well-retrieved short context. Context-compression techniques surveyed in Section 5—AttentionRAG [34] (30–50% context reduction), MobileRAG's Selective Content Reduction [99] (40–55% reduction with <1% accuracy loss), and SWE-Pruner [126]—can be understood as sharpening this parsimony further. Taken together, RAG and its context-compression extensions constitute a sustainability strategy in their own right, operating at the *algorithmic* rather than the hardware or scheduling layer.

Dynamic Voltage and Frequency Scaling (DVFS). Hardware-level power management has long been used in mobile processors, but applying DVFS to LLM inference requires awareness of the distinct computational phases. GreenLLM [86] addresses this by separating the prefill and decode stages and applying per-phase frequency optimization. During the compute-intensive prefill phase, GreenLLM maintains high clock frequencies to meet latency service-level objectives (SLOs). During the memory-bound decode phase, it reduces GPU/NPU frequency with minimal impact on token generation throughput, since the processor would otherwise stall on memory accesses. This phase-aware DVFS strategy achieves up to 34% energy reduction on representative inference workloads while meeting strict latency targets—a substantial saving that compounds across millions of daily queries in production deployments.

Carbon-aware scheduling. The carbon intensity of electricity generation varies by an order of magnitude across regions and fluctuates significantly within a single day as renewable generation rises and falls. Carbon-aware scheduling exploits this variability by routing inference workloads to data centers or time windows with lower grid carbon intensity. For latency-tolerant tasks (*e.g.*, background summarization, document indexing), deferring computation by even a few hours can substantially reduce emissions. For geographically distributed edge deployments, carbon-aware routing directs queries to devices or edge servers in regions currently powered by cleaner electricity. This strategy is particularly effective for hybrid edge-cloud architectures, where the cloud fallback can be selectively invoked based on both computational necessity and carbon cost.

At data-center scale, Zhang *et al.* [139] formulate this coupling as a mixed-integer program that jointly schedules rigid training jobs, routes elastic inference, dispatches local generation and battery storage, and manages grid interaction under latency and carbon-budget constraints. Their results show that joint compute–power optimization improves operational value and reduces emissions relative to decoupled compute-only or energy-only baselines; for hybrid edge–cloud RAG, the same principle argues that cloud fallback decisions should account for both workload flexibility and the power system serving the selected site.

Carbon-aware model adaptation. Beyond scheduling, carbon awareness can inform model selection and configuration. Carbon-Aware LoRA Scaling [142] proposes a multi-objective reinforcement learning framework for edge devices that dynamically adjusts the rank of LoRA adapters based on the current carbon intensity of the local grid. When carbon intensity is low, the system uses higher-rank adapters (more parameters, higher accuracy, more energy); when carbon intensity peaks, it scales down to lower-rank adapters that trade modest accuracy for significant energy savings. This approach operationalizes the insight that the "optimal" model configuration is not static but depends on the real-time environmental context of deployment.

AI-CARE [1] formalizes this trade-off through carbon-performance Pareto frontiers. For a given task, AI-CARE evaluates a family of models (*e.g.*, different sizes, quantization levels, adapter ranks) and plots their accuracy against carbon cost per query, identifying the Pareto-optimal set of configurations. Practitioners can then select the configuration that best matches their accuracy requirements and carbon budget. This framework makes explicit a trade-off that is typically implicit: every accuracy improvement has a carbon cost, and AI-CARE provides the analytical machinery to navigate this trade-off rationally.

Carbon-frugal search and retrieval. Sustainability considerations extend beyond the generation phase to the retrieval component of RAG pipelines. GaiaFlow [39] introduces semantic-guided diffusion tuning for carbon-frugal search, demonstrating that retrieval systems can be explicitly optimized to minimize carbon emissions per query while maintaining result quality. By incorporating carbon cost into the search objective function, GaiaFlow achieves comparable retrieval accuracy at 25–40% lower energy consumption compared to standard dense retrieval, suggesting that sustainability-aware retrieval design is a viable path toward greener RAG systems.

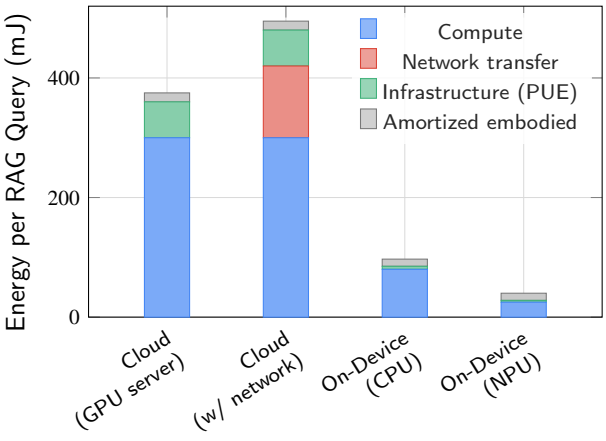


Figure 8: Decomposition of per-query energy cost across four deployment configurations for a 3B-parameter SLM RAG pipeline. Cloud inference incurs substantial infrastructure (PUE) and network transfer overhead absent in on-device settings. NPU execution reduces compute energy by ∼3× compared to on-device CPU. *These are representative estimates synthesized from [40, 99, 120], not direct measurements*; because the source studies use different hardware, models, and workloads, the values illustrate the relative composition of per-query energy across deployment types rather than absolute figures for any single system.

Accuracy–energy Pareto analysis. More broadly, the Green AI framework of Schwartz *et al.* [106] advocates for reporting the accuracy–efficiency Pareto frontier rather than single-point accuracy metrics. In the context of on-device RAG, this means evaluating system configurations along multiple dimensions—model size, quantization level, retrieval depth, context window—and identifying configurations that achieve near-optimal accuracy at dramatically lower energy cost. Empirically, diminishing returns are steep: moving from a 4-bit quantized 1B-parameter SLM to a full-precision 7B model may improve RAG answer quality by only 3–5% while increasing energy consumption by an order of magnitude. Making these trade-offs visible through systematic Pareto analysis empowers practitioners to make informed deployment decisions.

6.3. On-Device vs. Cloud: A Carbon Comparison

A central question for sustainable AI deployment is whether processing a query locally on the user's device or offloading it to a cloud data center results in lower total carbon emissions. The answer depends on a complex interplay of factors that we analyze below.

Cloud inference costs. Data center inference carries three principal energy overheads beyond the computation itself. First, *infrastructure overhead* captured by PUE adds

10–60% to raw compute energy for cooling, power conversion, networking equipment, and lighting. Hyperscale operators (*e.g.*, Google, Microsoft) achieve PUE values near 1.1, but enterprise and colocation facilities often operate at 1.4–1.6 [111]. Second, *network transfer energy*—the cost of transmitting the query from the user's device to the data center and returning the response—is frequently overlooked but non-trivial. For RAG workloads, this includes transmitting the query, receiving retrieved passages (potentially several kilobytes), and streaming the generated response. Estimates suggest network energy consumption of 0.1–0.5 kWh per gigabyte transferred, depending on access technology and network distance. It is worth being precise about the magnitude this implies: at that rate, a purely textual RAG exchange of order 10 kB costs only ∼1–5 mWh per query, so network transfer is negligible beside the 1–10 Wh of a cloud inference itself. Network energy becomes material only when the payload grows by three orders of magnitude—multi-modal queries, or architectures that ship whole documents rather than retrieved spans—in which case 1–10 MB per query translates to the ∼0.1–5 Wh range reported in Table 6 (the identity Wh = payload in MB × rate in kWh/GB follows because the MB-to-GB and kWh-to-Wh conversions cancel). Third, cloud inference requires the data center server to be *provisioned and idle* between requests, consuming baseline power that is amortized across queries but can dominate total energy at low utilization rates.

On-device inference costs. On-device inference eliminates network and infrastructure overhead but introduces its own costs. The device's battery or mains power supplies the computation directly, with effectively no PUE overhead. However, on-device inference introduces *embodied carbon* considerations: the manufacturing of smartphones, tablets, and edge devices is carbon-intensive, and attributing a fraction of this embodied carbon to AI workloads is methodologically complex. CO2-Meter [40] proposes an amortization approach that allocates embodied carbon proportional to the fraction of device lifetime spent on inference tasks. For typical usage patterns (a few dozen queries per day on a device with a 3–5 year lifespan), the amortized embodied carbon per query is small but non-negligible.

The crossover point. The relative carbon efficiency of on-device versus cloud inference depends on several factors:

- **Query frequency**: At low query volumes, the per-query amortization of cloud server provisioning and on-device embodied carbon both inflate costs. At high volumes, on-device inference amortizes embodied carbon effectively while avoiding repeated network round-trips.
- **Model size**: Smaller models (<3B parameters) run efficiently on-device with low energy per query, whereas larger models may require cloud offloading or model partitioning that negates efficiency gains.
- **Network conditions**: Poor connectivity increases latency and retransmission energy, favoring local inference. Conversely, high-bandwidth, low-latency connections reduce the marginal cost of cloud offloading.
- **Grid carbon intensity**: When the device is charged from a high-carbon grid and the cloud data center runs on renewable energy (or vice versa), the carbon calculus shifts accordingly. The growing adoption of renewable energy by hyperscale cloud providers complicates naïve assumptions that local is always greener.

As a rough heuristic—a rule of thumb rather than a measured threshold—on-device inference becomes carbon-favorable when (i) the model fits comfortably in device memory at a quantized precision, (ii) query frequency exceeds roughly 10–20 queries per day (the order of magnitude at which per-query amortization of on-device embodied carbon becomes small relative to the per-query operational and provisioning carbon of the cloud alternative; the exact crossover is deployment-specific and sensitive to device lifetime, model size, and both grids' carbon intensity), and (iii) the local grid carbon intensity is not dramatically higher than the cloud provider's. For the RAG use case specifically, keeping the retrieval index local eliminates the most data-intensive network transfer, further tilting the balance toward on-device deployment.

Privacy co-benefits. Beyond direct carbon savings, on-device inference eliminates data transmission entirely, yielding compound benefits. Network energy savings—on the order of 1–5 mWh for a text-only exchange, rising to the 0.1–5 Wh range only when multi-modal or document-scale payloads are shipped (consistent with Table 6)—accrue alongside privacy preservation, since neither the query nor the retrieved documents leave the device. This co-benefit is particularly salient in healthcare, legal, and financial applications where data sensitivity constrains architectural choices. The energy saved by avoiding encryption, TLS handshakes, and multi-hop routing further strengthens the case for local processing [120].

Fig. 8 decomposes per-query energy across four deployment configurations, illustrating how compute, network, infrastructure, and embodied carbon contributions shift between cloud and on-device settings. Table 6 presents a structured comparison of cloud and on-device inference across key sustainability dimensions.

Toward lifecycle-aware deployment decisions. The comparison above underscores that neither cloud nor on-device inference is universally greener. A lifecycle-aware perspective—one that accounts for operational energy, embodied carbon, network transfer, and grid carbon intensity—is necessary for principled deployment decisions. We revisit this theme in Section 7, where we integrate green AI metrics into a unified evaluation framework for sustainable on-device RAG systems.

Table 6
Cloud vs. On-Device Inference: A Sustainability Comparison

Metric	Cloud inference	On-device inference	Notes
Energy per query	0.001–0.01 kWh	0.00001–0.001 kWh	On-device is 10–100× lower for SLMs; gap narrows for larger models
PUE overhead	1.1–1.6×	≈1.0×	Data center cooling, power conversion, networking add 10–60% overhead
Network energy	0.001–0.005 Wh per query (text); 0.1–5 Wh (media)	0 (fully local)	At 0.1–0.5 kWh/GB, a text-only RAG exchange of ∼10 kB costs ∼1–5 mWh; the 0.1–5 Wh range applies only to multimodal or document-shipping workloads (1–10 MB)
Embodied carbon	Amortized across millions of users	Amortized over single device lifetime	Cloud amortizes well at scale; on-device depends on query frequency
Carbon intensity control	Provider-selected grid mix	User-dependent (local grid)	Hyperscale clouds increasingly use renewables; user grid varies widely
Latency	50–500 ms (network-dependent)	10–100 ms (compute-only)	Eliminates network round-trip; critical for interactive applications
Privacy	Data leaves device	Data stays on-device	On-device avoids transmission energy *and* preserves data privacy
Scalability	Elastic (pay-per-query)	Fixed (device-limited)	Cloud scales to arbitrary load; on-device is constrained by hardware

7. The Convergence: Toward Sustainable On-Device RAG

The preceding three sections have examined model compression (Section 4), on-device RAG architectures (Section 5), and Green AI measurement and strategies (Section 6) as largely independent research threads. In practice, however, every on-device RAG deployment is a *joint* instantiation of decisions drawn from all three domains: the quantization scheme determines both the memory headroom available for the retrieval index and the energy cost per generated token; the retrieval depth governs both answer quality and the number of prefill tokens that must be processed under a thermal envelope; and the sustainability posture—whether the system targets minimal energy, minimal carbon, or maximal battery life—constrains the feasible region of the design space. This section synthesizes these interdependencies into a unified framework. We first formalize the multi-dimensional trade-off space (Section 7.1), then propose the ALEMC evaluation framework that extends conventional accuracy metrics with explicit energy and carbon dimensions (Section 7.2), illustrate the framework through domain-specific case studies (Section 7.4), and distill recurring integration patterns that practitioners can adopt directly (Section 7.5).

7.1. Design Space and Trade-off Taxonomy

Every on-device RAG system operates within a four-dimensional trade-off space defined by *accuracy*, *latency*, *energy*, and *memory*. No single configuration dominates across all four dimensions; instead, each architectural and algorithmic decision shifts the system's position along a multi-dimensional Pareto frontier. Understanding this trade-off structure is essential for principled system design.

The four-way trade-off. We formalize the design space as follows. Let $\mathcal{A}$ denote task accuracy (*e.g.*, F1, exact match, ROUGE, or faithfulness), $\mathcal{L}$ denote end-to-end latency (time from query submission to complete response), $\mathcal{E}$ denote energy consumption per query (Joules), and $\mathcal{M}$ denote peak memory footprint (bytes). A system configuration θ drawn from the space of all feasible designs Θ maps to a point $(\mathcal{A}(\theta), \mathcal{L}(\theta), \mathcal{E}(\theta), \mathcal{M}(\theta))$ in this four-dimensional objective space. The Pareto-optimal set Θ^* consists of configurations for which no other configuration improves one objective without degrading at least one other:

$$\Theta^* = \{\theta \in \Theta : \nexists\, \theta' \in \Theta \text{ s.t. } \theta' \succ \theta\} \tag{1}$$

where $\theta' \succ \theta$ denotes that θ' weakly dominates θ across all four objectives and strictly improves at least one.

The configuration space Θ is the Cartesian product of decisions spanning all three pillar domains:

- **Compression decisions**: quantization bit-width (FP16, INT8, INT4, ternary), pruning sparsity level and pattern (unstructured, N:M structured), LoRA adapter rank, KV cache precision.
- **RAG pipeline decisions**: embedding model size, retrieval algorithm (dense, sparse, hybrid, graph-based), number of retrieved passages k, context compression ratio, reranker presence and complexity.
- **Sustainability decisions**: DVFS policy (fixed high, phase-aware, carbon-adaptive), NPU vs. CPU/GPU

execution, batch scheduling, carbon-aware adapter selection.

Cross-domain coupling. These decisions are not independent. Quantizing the SLM to 4-bit (a compression decision) frees approximately 1–2 GB of memory, which can be reallocated to a larger vector index (a RAG decision), in turn enabling retrieval over a richer corpus that improves accuracy. However, the additional retrieval results increase the prefill context length, raising both latency and energy consumption. Conversely, applying aggressive context compression via SparseRAG [146] or SRAS [96] (RAG decisions) reduces the prefill token count, lowering energy per query—but may discard relevant passages, degrading accuracy. Applying phase-aware DVFS [86] during the memory-bound decode phase (a sustainability decision) reduces energy without affecting latency when the processor is already memory-stalled, but DVFS applied during the compute-bound prefill phase will degrade latency. These cross-domain couplings mean that optimizing within a single pillar while holding the others fixed is fundamentally suboptimal.

The retrieval–energy tension. A particularly consequential coupling exists between retrieval depth and energy cost. Increasing the number of retrieved passages k improves answer accuracy by broadening the evidence base, reducing the risk of missing relevant content (failure point FP1 of Barnett *et al.* [5]) and strengthening the generator's grounding. However, each additional passage inflates the input context, increasing prefill computation quadratically (for standard attention) or linearly (for linear-complexity variants), and expanding the KV cache that must be stored in memory and read during every decode step. Empirically, the relationship between k and accuracy follows a pattern of diminishing returns—moving from k=1 to k=5 yields substantial accuracy gains, while moving from k=5 to k=20 yields marginal improvements that may even turn negative due to the "lost in the middle" phenomenon [85]. Energy, by contrast, scales approximately linearly with context length through the decode phase. The practitioner therefore faces a fundamental tension: *more retrieval improves accuracy but increases energy consumption; less retrieval saves energy but risks hallucination and incomplete answers.*

A decision framework for practitioners. Given the high dimensionality of the configuration space, we propose a constraint-driven decision framework that anchors on the binding constraint for a given deployment and selects the remaining parameters to optimize the non-binding objectives:

1. **Memory-constrained** (*e.g.*, IoT gateway with 2 GB RAM): Apply 4-bit quantization with quantized KV cache (INT4); use a graph-based index (MiniRAG [33]) that avoids storing dense embeddings; limit $k \leq 3$; select a sub-1B SLM (MobileLLM [89] or TinyLlama [140]).
2. **Latency-constrained** (*e.g.*, interactive voice assistant with <2 s response time): Use speculative RAG [128] to parallelize draft generation across document subsets; pair a 4-bit quantized SLM with NPU execution via MobileQuant [115]; apply SparseRAG [146] block-wise sparsity to prune irrelevant context before decoding; employ FlashAttention-derived tiled attention [26] to maximize prefill throughput.
3. **Energy-constrained** (*e.g.*, battery-powered mobile device with sustainability target): Apply aggressive context compression (MobileRAG SCR [99] or SRAS [96] document selection) to minimize prefill tokens; activate phase-aware DVFS [86] to reduce decode-phase power; use carbon-aware LoRA scaling [142] to lower adapter rank when battery is depleted or grid carbon intensity is high; prefer NPU execution at INT8 precision where TOPS/watt is maximized.
4. **Accuracy-constrained** (*e.g.*, medical or legal domain requiring >95% faithfulness): Deploy the largest SLM that fits the memory budget at the highest feasible precision (INT8 via SmoothQuant [131]); use hybrid BM25 + dense retrieval with a lightweight cross-attention reranker (PocketRAG [67]); retrieve k=5–10 passages; accept higher energy and latency as the cost of accuracy assurance.

This framework does not prescribe a single "best" configuration but rather makes the trade-off structure explicit, enabling practitioners to navigate the Pareto frontier guided by their deployment-specific constraints. Formal multi-objective optimization methods can assist in this navigation: NeuroPareto [38] proposes calibrated acquisition functions for costly many-goal search in vast parameter spaces, providing a principled Bayesian approach to exploring the accuracy–latency–energy–memory Pareto surface without exhaustively evaluating all configurations. Such methods are directly applicable to the ALEMC design space, where each configuration evaluation requires running inference on real hardware and measuring multiple objectives simultaneously. Figure 9 visualizes representative configurations from the literature on the accuracy–energy plane, illustrating how compression, retrieval, and sustainability decisions collectively shape the frontier.

A note on cross-study data synthesis. Several figures in this survey (notably Fig. 8 and Fig. 9) aggregate numbers reported by different primary studies. We stress that these are *syntheses for intuition, not head-to-head measurements*, for three reasons. First, the source studies run on heterogeneous hardware (mobile SoCs, edge GPUs, and datacenter accelerators), different SLM backbones and quantization schemes, and different retrieval corpora and query distributions, so their absolute figures are not directly comparable; we use them only to convey relative composition and order-of-magnitude structure. Second, accuracy is reported inconsistently across the literature—some studies report token-level F1, others exact match, others task-specific correctness or faithfulness. Where we place multiple systems on a shared

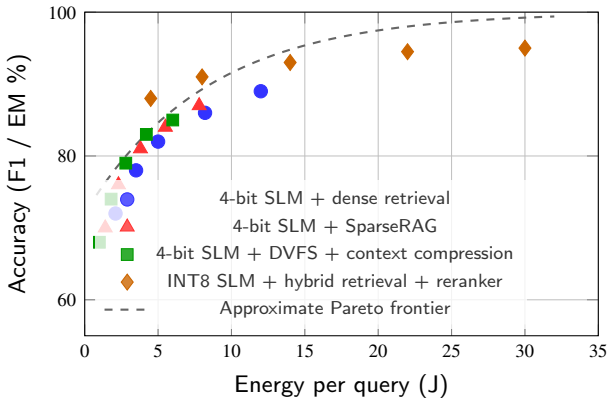


Figure 9: *Illustrative* accuracy–energy Pareto frontier for on-device RAG configurations. Each point represents a system configuration defined by compression, retrieval, and sustainability choices. Points closer to the upper-left corner are more efficient. The dashed curve approximates the Pareto frontier; configurations below this curve are dominated. Coordinates are *synthesized estimates*, not direct measurements: they are drawn from reported results in [99, 33, 146, 67], which use differing hardware and datasets and are therefore not strictly commensurable. For a *measured* accuracy–energy frontier obtained on a single device under a fixed pipeline, see Fig. 11 in the empirical study (Section 7.3).

accuracy axis, we treat these as approximately commensurable quality scores on a 0–100% scale rather than converting between metrics, and we avoid ranking systems by small accuracy differences that fall within this metric-mismatch noise. Third, where a study does not report a dimension we need (*e.g.*, energy for a system that reported only latency and memory), we mark the value as an estimate rather than silently imputing it. These limitations are precisely what motivates the controlled, single-pipeline empirical study of Section 7.3, in which every dimension is measured on the same corpus and query set so that the resulting numbers *are* mutually comparable.

7.2. A Proposed Evaluation Framework: ALEMC

Existing RAG evaluation focuses almost exclusively on answer quality [32], while on-device benchmarks report latency and memory but rarely energy or carbon. This fragmented evaluation landscape makes it impossible to compare systems holistically or to optimize for sustainability. The broader AI-infrastructure community has reached a similar conclusion: He [50] proposes a cross-layer metric taxonomy that jointly integrates performance, efficiency, and cost dimensions across the AI stack, arguing that single-pillar evaluations systematically misrepresent deployment reality. We extend this line of thinking to the on-device RAG setting. We propose the **ALEMC** framework—**A**ccuracy, **L**atency, **E**nergy, **M**emory, **C**arbon—as a unified evaluation protocol for on-device RAG systems.

7.2.1. *Metric Definitions*

We define each ALEMC dimension with its formal metric, unit, recommended measurement methodology, and suggested target ranges for on-device deployment.

Accuracy ($\mathcal{A}$). Accuracy encompasses both task-specific performance and retrieval-grounded faithfulness. For extractive question answering, we adopt exact match (EM) and token-level F1; for abstractive generation, ROUGE-L; for open-ended responses, we incorporate the RAGAS [32] decomposition into *context relevance* (fraction of retrieved context that is pertinent to the query), *answer faithfulness* (fraction of generated claims supported by the retrieved context), and *answer relevance* (degree to which the response addresses the query). The composite accuracy score is the harmonic mean of task performance and faithfulness:

$$\mathcal{A} = \frac{2 \cdot \text{TaskScore} \cdot \text{Faithfulness}}{\text{TaskScore} + \text{Faithfulness}} \tag{2}$$

where $\text{TaskScore} \in \{\text{F1}, \text{EM}, \text{ROUGE-L}\}$ depending on the task. This formulation penalizes systems that achieve high task scores through hallucination (*i.e.*, high F1 but low faithfulness) or that are faithful but uninformative (*i.e.*, high faithfulness but low F1).

Latency ($\mathcal{L}$). We distinguish two latency components. *Time-to-first-token* (TTFT) measures the delay from query submission to the generation of the first output token, encompassing query encoding, retrieval, optional reranking, context assembly, and prefill computation. For interactive applications, TTFT governs perceived responsiveness and should remain below 500 ms. *End-to-end latency* ($\mathcal{L}_{\text{e2e}}$) measures the total time to produce a complete response, including the full autoregressive decode phase. We report both metrics, as TTFT is critical for user experience while $\mathcal{L}_{\text{e2e}}$ determines throughput:

$$\mathcal{L}_{\text{e2e}} = t_{\text{retrieval}} + t_{\text{prefill}} + n_{\text{tokens}} \cdot t_{\text{decode}} \tag{3}$$

where $t_{\text{retrieval}}$ is the time for vector search and optional reranking, t_{prefill} is the prefill latency over the concatenated context, n_{tokens} is the number of generated tokens, and t_{decode} is the per-token decode latency.

Energy ($\mathcal{E}$). Energy is measured in Joules per query and represents the total electrical energy consumed across all hardware components (NPU, CPU, GPU, memory subsystem) during a single RAG query–response cycle. We recommend measurement via CO2-Meter [40] for edge devices, which provides phase-decomposed estimates (prefill energy vs. decode energy), or EnviroLLM [3] for local LLM deployments with real-time power monitoring. When hardware power meters are unavailable, software estimation via CodeCarbon [23] using Intel RAPL or platform-specific power APIs provides a reasonable approximation with 10–20% uncertainty. The energy metric should be reported as the mean and standard deviation over at least 100 queries to account for thermal throttling and DVFS state variability:

$$\mathcal{E} = \int_{t_0}^{t_0+\mathcal{L}_{\text{e2e}}} P(t)\,\mathrm{d}t \tag{4}$$

where $P(t)$ is the instantaneous system power draw at time t.

Memory ($\mathcal{M}$). Memory is reported as peak resident set size (RSS) during query processing, in megabytes. This encompasses four principal components: model weights (M_w), KV cache (M_{kv}), embedding model (M_{emb}), and vector index (M_{idx}):

$$\mathcal{M} = M_w + M_{\text{kv}} + M_{\text{emb}} + M_{\text{idx}} + M_{\text{overhead}} \tag{5}$$

where M_{overhead} captures runtime buffers, activations, and operating system overhead. For on-device deployment, $\mathcal{M}$ must fit within the device's available application memory, which is typically 3–6 GB on smartphones after accounting for the operating system and background processes.

Carbon ($\mathcal{C}$). Carbon intensity is expressed in grams of CO_2-equivalent per query (gCO_2e/query) and is derived from the energy metric by incorporating the local electricity grid's carbon intensity factor γ (gCO_2/kWh):

$$\mathcal{C} = \mathcal{E} \times \gamma \times 10^{-3}/3600 \tag{6}$$

where the conversion factor transforms Joules and gCO_2/kWh into gCO_2e/query. The grid carbon intensity γ varies by geography and time of day—from under 20 gCO_2/kWh (Norway, hydroelectric) to over 700 gCO_2/kWh (Poland, coal)—and should be reported alongside all carbon measurements to enable cross-study comparison. For lifecycle-complete assessment, amortized embodied carbon $\mathcal{C}_{\text{embodied}}$ can be added following the methodology of CO2-Meter [40].

7.2.2. Composite Score

To enable single-number comparison when needed, we propose a weighted geometric mean over normalized ALEMC dimensions. We state the score in a single *goodness* form, in which every dimension is first mapped to a "higher-is-better" quantity on a common scale and all five exponents are positive:

$$S_{\text{ALEMC}} = \prod_{i \in \{a,l,e,m,c\}} g_i^{w_i}, \qquad \sum_i w_i = 1 \tag{7}$$

where the goodness g_i of dimension i over the set of compared systems is

$$g_i = \delta + (1-\delta)\,\hat{g}_i,$$
$$\hat{g}_i = \begin{cases} n_i & \text{(accuracy)} \\ 1 - n_i & \text{(latency, energy, memory, carbon)} \end{cases} \tag{8}$$

and $n_i = (v_i - v_i^{\min})/(v_i^{\max} - v_i^{\min}) \in [0,1]$ is the min–max normalization of the raw value v_i across the compared set. The weights $w_i \geq 0$ allow practitioners to customize the score to their deployment priorities. The geometric-mean formulation (as opposed to an arithmetic mean) ensures that unacceptable performance on any single axis cannot be masked by strong performance on others.

We stress that Eq. 7 is the *sole* definition of the composite. An earlier formulation of this framework expressed the cost dimensions as negative exponents on their normalized values, $\hat{\mathcal{L}}^{-w_l}\hat{\mathcal{E}}^{-w_e}\hat{\mathcal{M}}^{-w_m}\hat{\mathcal{C}}^{-w_c}$. That form is *not* algebraically equivalent to Eq. 7—$n^{-w} \neq (\delta + (1-\delta)(1-n))^w$—and it is also ill-behaved: the best-in-set system on a cost dimension normalizes to $n = 0$, and 0^{-w} diverges. We therefore adopt the goodness form throughout, and all reported scores are computed from it.

Boundary handling (δ-smoothing). Mapping each dimension into $[\delta, 1]$ rather than $[0, 1]$, with a small floor δ (we use $\delta = 0.05$), keeps the score finite and well behaved at both extremes. Without the floor, a single worst-in-set value ($\hat{g}_i = 0$) would zero out the entire product regardless of performance on the other four dimensions. The floor instead caps the penalty a single worst-in-set dimension can impose at δ^{w_i}, which we found essential for producing stable rankings in the empirical study of Section 7.3.

Zero-range dimensions. When every compared system takes the same value on a dimension ($v_i^{\max} = v_i^{\min}$), the min–max map is undefined. We define $g_i = 1$ for all systems in that case, so a dimension on which nothing distinguishes the candidates contributes a neutral factor of 1 rather than a division by zero or an arbitrary tie-break.

Energy–carbon redundancy. Eq. 6 makes carbon a fixed multiple of energy at a given grid intensity γ. Consequently, whenever a comparison is conducted at a *single* fixed γ—as in most single-site evaluations, including our own study below—$\mathcal{E}$ and $\mathcal{C}$ produce *identical* normalized goodness values, and scoring both would silently apply the combined weight $w_e + w_c$ to one underlying signal. We therefore adopt the following rule: under a fixed γ, fold w_c into w_e and score four dimensions; retain carbon as an independently scored dimension only when γ genuinely varies across the compared systems, as in multi-region deployments, carbon-aware scheduling, or fleets spanning different grid mixes. Carbon remains a mandatory *reporting* dimension in all cases—the redundancy concerns only its role in the composite, not its role in the evaluation.

Reference-set sensitivity. Because min–max normalization is relative to the set of compared systems, adding or removing a system rescales every dimension and can, in principle, re-order the others. Practitioners should therefore (i) report the composite alongside the raw per-dimension values (which are set-independent), and (ii) where longitudinal comparability matters, normalize against fixed anchors (*e.g.*, the target-range bounds in Table 7) rather than the compared set. We quantify this effect empirically in Section 7.3 via a drop-one-system analysis.

Default weight profiles for common deployment scenarios include: *accuracy-first* (w_a=0.40, w_l=0.20, w_e=0.15, w_m=0.10, w_c=0.15) for medical and legal applications; *efficiency-first* (w_a=0.15, w_l=0.15, w_e=0.25, w_m=0.25, w_c=0.20) for always-on IoT devices, where memory headroom is as binding as energy; and *balanced* (w_a=0.25, w_l=0.20, w_e=0.20, w_m=0.15, w_c=0.20) for general mobile deployment.

Table 7
The ALEMC Evaluation Framework for Sustainable On-Device RAG Systems

Dimension	Metric	Unit	Measurement tool	Description	Target range
Accuracy ($\mathcal{A}$)	F1 / EM / ROUGE-L + Faithfulness	% (harmonic mean)	RAGAS [32], HELM [82]	Harmonic mean of task score and RAGAS faithfulness	$\geq 80\%$
Latency ($\mathcal{L}$)	TTFT; $\mathcal{L}_{e2e}$	ms	System timer (wall-clock)	Time-to-first-token and end-to-end query-to-response	TTFT < 500 ms; $\mathcal{L}_{e2e} < 5$ s
Energy ($\mathcal{E}$)	Joules per query	J	CO2-Meter [40], EnviroLLM [3]	Total energy (NPU + CPU + DRAM) integrated over query duration	< 10 J (mobile); < 30 J (edge)
Memory ($\mathcal{M}$)	Peak RSS	MB	OS memory profiler	Model weights + KV cache + embedding model + vector index	< 4 GB (mobile)
Carbon ($\mathcal{C}$)	gCO_2e per query	gCO_2e	$\mathcal{E} \times \gamma$	Operational carbon; optionally add amortized embodied carbon	$< 0.05\, gCO_2e$

On what profile comparisons can and cannot show. A caution is warranted about using a small set of fixed profiles as evidence of robustness. If a system is *extremal*—best-in-set or worst-in-set—on every dimension except one, its composite score reduces to δ^{w_j} (or 1) where j is the single non-extremal dimension, and it is therefore *invariant* to every weight except w_j by construction. A set of profiles that happens to assign the same w_j will then return an identical score for that system under all profiles, which reflects the arithmetic of Eq. 7 rather than any genuine insensitivity to weighting. For this reason the profiles above deliberately vary w_m over $[0.10, 0.25]$, and—more importantly—we report weight sensitivity in Section 7.3 as a *continuous sweep* that identifies the threshold at which the top-ranked system changes, rather than as a comparison of three hand-picked points. A crossover threshold is a falsifiable statement about robustness; three coincident scores are not.

7.2.3. The ALEMC Framework Summary

Table 7 presents the complete framework with recommended measurement tools and target ranges for on-device RAG deployment.

7.2.4. Benchmark Protocol

To ensure reproducibility and comparability across studies, we propose a standardized benchmark protocol for ALEMC evaluation:

1. **Hardware standardization.** Evaluate on at least one reference mobile SoC (*e.g.*, Qualcomm Snapdragon 8 Gen 3 or Apple A17 Pro) under controlled thermal conditions (ambient 22–25°C, device not thermally pre-loaded). Report the SoC model, NPU utilization, and thermal state at measurement time.
2. **Dataset standardization.** Use a common QA benchmark subset—we recommend 500 questions from Natural Questions (open-domain, factoid) and 500 from HotpotQA (multi-hop reasoning)—with a fixed retrieval corpus of 100,000 passages derived from Wikipedia. This ensures consistent retrieval difficulty across systems.
3. **Energy measurement.** Report energy per query as the mean ± standard deviation over $N \geq 100$ queries, with a 5-query warm-up to stabilize thermal state. Measure using hardware power monitors where available; when using software estimation, report the tool and version.
4. **Carbon reporting.** Report carbon using at least three reference grid intensities: low (50 gCO_2/kWh, representing renewable-heavy grids), medium (300 gCO_2/kWh, representing mixed grids), and high (600 gCO_2/kWh, representing fossil-heavy grids). This enables readers to interpolate to their deployment geography.
5. **Reproducibility.** Publish model weights, quantization configurations, retrieval indices, and measurement scripts. Report all five ALEMC dimensions; partial reporting undermines the framework's utility.

Benchmark size should also be justified before results are observed. Zhuang *et al.* [147] adapt a paired minimum-detectable-effect calculation to 4-bit quantization benchmarks, showing how the item count and FP16–NF4 disagreement rate determine which accuracy differences a study can reliably detect. Their pilot audit further finds that prompt-template variation can match or exceed observed quantization deltas, motivating pre-registration of both the paired sample budget and prompt configuration when ALEMC comparisons assess compressed on-device models.

Fig. 12 applies the ALEMC framework to the four on-device RAG systems reviewed in Section 5, visualizing each system's profile across all five dimensions on a normalized radar chart. The visualization reveals characteristic trade-off signatures: PocketRAG's accuracy-first posture (driven by its cross-attention reranker) comes at the cost of energy and carbon efficiency, while MobileRAG's holistic optimization stack yields the strongest latency and energy profile but does not invest in reranking for maximum accuracy.

A practical concern for any evaluation framework is the reliability of the accuracy dimension, particularly when LLM-based judges are used. Li *et al.* [74] systematize the LLM-as-a-judge paradigm across what is judged, how judgment is performed, and how judge systems are benchmarked, while identifying bias, vulnerability, and dynamic evaluation as persistent limitations. Recent work on robust statistical evaluation of LLMs [35] further demonstrates that imperfect judges introduce systematic biases that can distort comparative rankings, and proposes statistical corrections that remain valid under judge noise. For the ALEMC framework, we recommend reporting accuracy using multiple evaluation methods (automated metrics, RAGAS decomposition, and human validation on a subset) to mitigate single-judge bias.

Configuration sensitivity presents a separate reproducibility risk. SafetyRepro shows that harness choices alone can reverse pairwise model rankings on alignment benchmarks [77], so benchmark reports should freeze and disclose decoding, prompting, scoring, and evaluator configurations rather than treating a rank order as configuration-invariant. Accuracy-only summaries also miss whether a reasoning trace recognizes biased input: Sun *et al.* [113] distinguish *susceptibility* from *acknowledgment* and find large acknowledgment differences between models with similar final-answer susceptibility. For safety-sensitive on-device RAG, ALEMC's accuracy dimension should therefore be accompanied by configuration-stability and trace-level bias diagnostics.

We note that existing on-device RAG studies do not report all five ALEMC dimensions simultaneously. MobileRAG [99] reports accuracy, latency, memory, and power consumption but not carbon. PocketRAG [67] reports accuracy, latency, and memory but not energy or carbon. MiniRAG [33] reports accuracy and memory but not latency, energy, or carbon. This fragmentation is precisely the gap that ALEMC aims to close. To demonstrate that the framework is not merely a recommendation but is measurable end-to-end, we instantiate it ourselves and report all five dimensions together for every configuration in the proof-of-concept study of Section 7.3; we are not aware of a prior on-device RAG evaluation that reports all five simultaneously, though we make no priority claim. By establishing energy and carbon as mandatory reporting dimensions alongside accuracy, latency, and memory, we seek to shift the evaluation culture toward holistic sustainability awareness.

7.3. Empirical Demonstration of ALEMC

To validate that the ALEMC framework works end-to-end, we conduct a scoped proof-of-concept study. It is designed to expose two ways in which holistic evaluation departs from single-metric evaluation: first, that measuring the right quantity can *reorder* systems relative to a single-dimension view (a phone that draws the least instantaneous power is the least energy-efficient *per query*; a configuration that maximizes retrieval faithfulness is Pareto-*dominated* on cost); and second, that the composite ranking is *sensitive to measurement choices*—the energy boundary and the memory accounting protocol—in ways that a single headline number renders invisible. This is a deliberately reduced instantiation of the benchmark protocol of Section 7.2 (a 200-question frozen set, a 12,000-passage corpus, single run per configuration), not a definitive benchmark; its purpose is to exercise all five dimensions on real hardware and surface exactly these effects. The instrumentation follows our own prior single-platform study of energy-efficient on-device RAG on a Snapdragon X Elite NPU laptop [19], which established the whole-device energy attribution and per-stage timing protocol reused here; the contribution of the present study is to place that protocol under the five-dimensional ALEMC lens and to run it across three hardware tiers, which is what exposes the reordering effects described above. All scripts, the frozen query set, the prebuilt index, and the raw per-query logs are released for reproducibility.[2]

Setup. The pipeline is identical across all runs: a 4-bit `Q4_K_M` Gemma-2-2B-it generator served by a locally built `llama.cpp` [43] (commit `6658925`), a `bge-small-en-v1.5` query embedder (ONNX via `fastembed` 0.8.0 on `onnxruntime` 1.27.0, no PyTorch dependency, enabling on-device execution), and brute-force dense retrieval (FAISS [64] 1.13.0, `IndexFlatIP`, k=5 unless varied) over a fixed 12,000-passage Wikipedia-derived corpus. Decoding is deterministic and identical on every backend: greedy sampling (temperature 0), a 256-token generation cap, a fixed prompt template, and a 5-query warm-up that is excluded from all reported statistics. The evaluation set comprises 200 questions—100 factoid (SQuAD) and 100 multi-hop (HotpotQA); we substitute SQuAD for Natural Questions so that every question's gold passage is present in the scoped corpus.

Measurement boundaries (and their limits). The three tiers are *not* instrumented at a common hardware boundary, and we report this explicitly rather than presenting the columns as directly commensurable. Energy is whole-device on both Snapdragon tiers—battery discharge via WMI on the X Elite laptop, and a fuel-gauge charge-counter over `adb` on the 8 Elite phone—but *GPU-package only* (NVML) on the RTX 5070 Ti desktop. The desktop figure therefore excludes CPU-side query embedding and FAISS retrieval, system DRAM, the motherboard, storage, cooling, and power-supply conversion losses. We report both a gross figure (E_{gross}, directly integrated over the query window) and an idle-subtracted figure (E_{net}, which isolates the inference-attributable component against a measured idle baseline); the two answer different questions and are not interchangeable, so Table 8 carries both. Memory was likewise originally captured under different protocols per tier, which is not defensible for cross-tier comparison; we now report a *harmonized* figure (M_{warm}: resident footprint after a short warm-up on a freshly started server, available for all three tiers) alongside the sustained-serving peak (M_{sust}), and use the harmonized figure for all rankings. GPU VRAM is not

[2] All code and data are openly available at https://github.com/zhycheng614/alemc-ondevice-rag (MIT-licensed code; CC BY-SA 4.0 data). Model weights are obtained separately as described in the repository.

attributed to process RSS under WDDM and is excluded from both.

Accuracy across backends. Query embedding, retrieval, and scoring are computed identically for every device, and answers are scored centrally (exact match, token-F1, and a lexical faithfulness proxy). Generation, however, *is* performed independently on each backend, so accuracy is not identical by construction: differing kernel implementations and floating-point accumulation orders across the CPU and CUDA builds change tokenization ties and therefore some generated strings. The measured values (0.550, 0.624, 0.630) are close enough to treat the tiers as delivering comparable answer quality, but the residual spread also reflects the phone's smaller completed sample (below), and we do not attribute the differences to the hardware itself.

Completed sample sizes. The frozen query set contains 200 questions, but the number actually completed differs per run and we report both n (scored) and n_E (with a valid energy window) per row in Tables 8 and 10. The desktop and laptop baselines completed 195/195. The phone tier was deliberately run on a 60-question subset (55 scored after warm-up, of which 32 yielded a usable energy window): a full 200-query pass costs roughly 92 minutes of sustained load on a handset, with attendant thermal throttling, and the `adb` charge-counter has a resolution of ∼10–30 mAh, so a substantial fraction of individual query windows register no measurable charge delta. The `C3` configuration completed 161 scored queries with 93 energy windows. The phone tier consequently carries materially weaker statistical support than the other two, and we flag this wherever its numbers are used. Full per-device instrumentation, version manifests, and the raw per-query logs are given in the released artifacts.

Experiment 1 (cross-tier): the race-to-idle effect. Table 8 reports the identical pipeline measured across three hardware tiers, and the headline result contradicts the naive intuition. The RTX 5070 Ti desktop draws by far the most instantaneous power—a mean of ≈ 98 W GPU-package over the run, and ≈ 105 W averaged over the query windows themselves, versus tens of watts on the CPU tiers—yet because it is roughly 45× faster end-to-end than the laptop and 240× faster than the phone (105 ms vs. 4,764 ms vs. 25,536 ms), its *per-query* energy (11.1 J gross) is the *lowest* of the three tiers. Conversely, the Snapdragon 8 Elite phone draws the least power but its CPU-bound prefill over the retrieved context makes it so slow that its per-query energy (423.9 J gross) is the *highest*. This is a "race-to-idle" effect, and it refutes the common "GPU = energy-hungry, phone = energy-frugal" heuristic at the per-query level. It is visible only because ALEMC reports energy *per query* rather than instantaneous power.

Because the tiers are instrumented at different boundaries, we must ask whether this ordering is an artifact of the desktop's narrower GPU-package accounting. It is not, and the margin is large enough to settle the question without a wall-plug meter. Even under a deliberately pessimistic whole-system attribution—charging the desktop a sustained 400 W, near the capacity of its power supply and roughly four times its measured GPU-package draw, for the entire query window—its per-query energy would be ≈ 42 J: still an order of magnitude below the phone's 423.9 J and 3.4× below the laptop's 144.2 J. A realistic whole-system figure (CPU, board, and conversion losses added to the measured GPU draw) lands nearer 25 J. The *ordering* desktop < laptop < phone therefore holds under any plausible accounting choice, even though the *absolute* per-query energies are boundary-dependent and should not be quoted across platforms without stating the boundary. Note also that the same conclusion follows from the idle-subtracted column: E_{net} is 7.6 J, 10.2 J, and 224.7 J respectively, preserving the ordering.

Experiment 1 (cross-tier): memory accounting decides the ranking. On accuracy, latency, energy, and carbon the desktop is best-in-set, so any single-dimension evaluation—and even an energy-*and*-carbon-only "green" comparison—ranks it first. Whether the *composite* agrees turns entirely on how memory is accounted for, and this is the more consequential finding of the experiment.

Under the harmonized short-warm-up protocol (Table 9, Panel A), the desktop's resident footprint is 2.7 GB against the laptop's 4.5 GB and the phone's 2.3 GB; it is not worst-in-set on any dimension and ranks first under all three weight profiles and across the entire w_m sweep. But the CUDA `llama-server`'s host resident set does not plateau: it grows from ≈ 2.0 GB when freshly loaded to ≈ 10.3 GB over sustained serving—a CUDA host-allocator and buffer-caching effect under WDDM, which we reproduced on an independently restarted server. Scored under that sustained-serving accounting (Panel B), the desktop becomes worst-in-set on memory, and the geometric mean demotes it below the laptop once w_m exceeds ≈ 0.085—a threshold well below any of our three profiles, so the laptop takes rank 1 under all of them.

The substantive lesson is therefore *not* that one device beats another. It is that a defensible-looking choice of measurement protocol—steady state after warm-up, versus peak under sustained load—silently determines the winner of a five-dimensional evaluation, while leaving four of the five dimensions untouched. This is the same class of hazard as the reference-set sensitivity discussed in Section 7.2, and it reinforces the same practitioner guidance: report the raw per-dimension values (which Table 8 retains, under both protocols) alongside any composite, and state the accounting protocol as part of the result. It also surfaces a genuine deployment risk that no single-dimension evaluation would expose—host-memory pressure that appears only under sustained load—which is precisely the kind of finding a multi-dimensional framework exists to catch.

Fig. 10 visualizes the measured profiles on the five ALEMC axes under the harmonized memory protocol, normalized so that a larger radius uniformly indicates better efficiency. The desktop reaches the frontier on four axes and is mid-set on memory; the phone's polygon collapses on everything except memory. Under sustained-serving accounting the desktop's memory radius would instead collapse to

Table 8
Experiment 1 (cross-tier): the identical RAG pipeline (Gemma-2-2B `Q4_K_M`, dense retrieval, k=5) measured across three hardware tiers. **Measurement boundaries differ across tiers and are stated explicitly**: the Snapdragon tiers are instrumented whole-device (battery), whereas the desktop reports GPU-package power only (NVML), excluding CPU-side embedding and retrieval, DRAM, board, cooling, and PSU losses. E_{gross} is the directly integrated figure; E_{net} subtracts the measured idle baseline to isolate the inference-attributable component. Memory is reported under two protocols: M_{warm} (harmonized short-warm-up steady state on a freshly started server, comparable across all tiers) and M_{sust} (peak under sustained serving; the desktop's CUDA host allocator grows to 10.3 GB, while the phone runs no persistent server and so cannot grow). n is the number of scored queries and n_E the number with valid energy windows. Carbon is derived from E_{gross} at the medium grid intensity (300 gCO_2/kWh) and is therefore a fixed multiple of it, not an independent measurement.

Tier / Device	Boundary	n / n_E	Accuracy	$\mathcal{L}_{\text{e2e}}$ (ms)	E_{gross} (J)	E_{net} (J)	M_{warm} (MB)	M_{sust} (MB)	Carbon (gCO_2e)
mobile / Snapdragon 8 Elite	whole-device	55 / 32	0.550±0.433	25536±5752	423.89±221.45	224.71±220.21	2339	2339	0.0353
edge / Snapdragon X Elite	whole-device	195 / 195	0.624±0.450	4764±915	144.22±27.40	10.22±3.34	4499	–	0.0120
desktop / RTX 5070 Ti	GPU package	195 / 195	0.630±0.449	105±18	11.09±2.17	7.57±1.61	2683	10316	0.0009

Table 9
ALEMC composite ranking (§7.2), δ=0.05. **The memory accounting protocol, not the weight profile, decides the cross-tier ordering.** Under the harmonized short-warm-up protocol (Panel A) the desktop is rank-1 under every profile and across the entire w_m sweep. Under sustained-serving accounting (Panel B), where the CUDA host allocator grows to 10.3 GB, the laptop takes rank-1 once w_m exceeds ≈ 0.085. Under the fixed grid intensity used here, carbon is a constant multiple of energy, so the two dimensions normalize identically; scoring both would double-count one signal. We therefore fold w_c into w_e and score four dimensions, while still *reporting* all five.

System	accuracy-first	efficiency-first	balanced
Panel A: harmonized warm-up memory (M_{warm})			
RTX 5070 Ti	#1 (0.984)	#1 (0.960)	#1 (0.976)
Snapdragon X Elite	#2 (0.620)	#2 (0.385)	#2 (0.520)
Snapdragon 8 Elite	#3 (0.067)	#3 (0.106)	#3 (0.078)
Panel B: sustained-serving memory (M_{sust})			
RTX 5070 Ti	#2 (0.741)	#2 (0.473)	#2 (0.638)
Snapdragon X Elite	#1 (0.811)	#1 (0.756)	#1 (0.780)
Snapdragon 8 Elite	#3 (0.067)	#3 (0.106)	#3 (0.078)

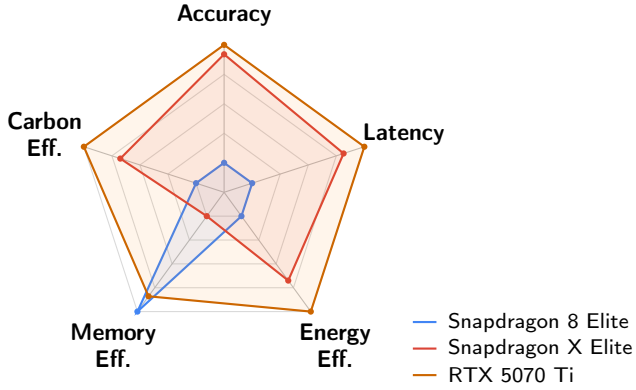


Figure 10: Measured ALEMC profile across the three hardware tiers (Experiment 1). Each axis is efficiency-normalized over the compared set so that a larger radius is better on every dimension (latency, energy, memory, and carbon are inverted). Memory uses the harmonized short-warm-up protocol (M_{warm} in Table 8); under sustained-serving accounting the desktop's memory radius collapses instead (Table 9, Panel B). RTX 5070 Ti reaches the frontier on accuracy, latency, energy, carbon and is not worst on any axis. Note that the axes are *set-relative*: they show each tier's standing within this three-way comparison, not an absolute score.

the minimum, which is the visual form of the ranking flip described above.

Experiment 2 (configuration axis). Holding hardware fixed (the Snapdragon X Elite laptop), we vary the RAG configuration across three settings *inspired by* the design archetypes of the surveyed systems: C1 (lightweight, k=3, no reranker), C2 (balanced, k=5 with context compression), and C3 (accuracy-oriented, k=10 with a cross-encoder-style reranking stage). We stress that these are one common Gemma/BGE/FAISS pipeline at three operating points, *not* reimplementations of MiniRAG, EdgeRAG, or PocketRAG; they are labelled generically throughout for that reason.

Table 10 and Fig. 11 report the result. End-task accuracy is *flat* across the three settings (0.586, 0.569, 0.583): the differences are far smaller than the per-query spread ($\sigma \approx 0.47$) and we draw no ordering from them. What separates the configurations is cost. C3 attains the highest retrieval faithfulness (0.835 versus 0.756 for C1) yet converts none of it into end-task accuracy, while consuming 3.6× the energy (412.3 J vs. 114.6 J) and 4.9× the latency of C1. C3 is therefore **Pareto-dominated by C1**: C1 matches its accuracy at a fraction of the cost. We emphasize that this dominance rests on the energy and latency axes, where the separation is large and unambiguous, and *not* on the 0.003 accuracy difference, which is noise. A single-dimension evaluation optimizing for retrieval faithfulness or retrieval depth would select C3; the ALEMC view exposes it as dominated. This is precisely the kind of misranking that holistic evaluation is meant to prevent. The composite ranks C1 > C2 > C3 under all three weight profiles, and C3 never attains rank 1 anywhere in the weight sweep.

Robustness. We probed the composite along three axes—weighting, reference set, and measurement protocol—and release all three analyses. Four findings bear on the framework's proper use.

Table 10
Experiment 2 (configuration axis, Snapdragon X Elite): one common Gemma/BGE/FAISS pipeline run at three settings *inspired by* the design archetypes of the surveyed systems—these are **not** reimplementations of MiniRAG, EdgeRAG, or PocketRAG. Accuracy is *statistically flat* across the three settings (all differences fall well inside the per-query spread), while energy and latency rise steeply: the accuracy–energy Pareto (Fig. 11) therefore exposes the accuracy-oriented setting as dominated on the *cost* axes. ± is the per-query standard deviation over n scored queries (n_E with valid energy windows); it describes spread across queries, not the precision of the mean.

Config	Setting	n / n_E	Accuracy	Energy (J)	$\mathcal{L}_{e2e}$ (ms)
C1	lightweight, k=3	194 / 194	0.586±0.467	114.58±39.77	2976±769
C2	balanced, k=5, compress	193 / 193	0.569±0.470	75.26±7.64	2209±252
C3	accuracy-first, k=10, rerank	161 / 93	0.583±0.467	412.27±104.33	14712±4661

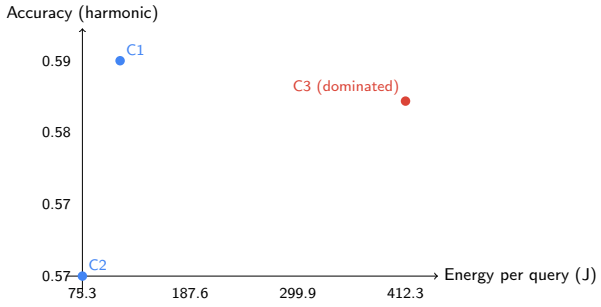


Figure 11: Measured accuracy–energy Pareto frontier for Experiment 2 (Snapdragon X Elite). The dominated configuration (red) is off the frontier: another configuration achieves at least equal accuracy at lower energy. Accuracy differences between these settings are small relative to the per-query spread (Table 10), so dominance here should be read as resting on the *energy* axis, where the separation is large and unambiguous, rather than on the accuracy ordering. Points are single-run point estimates.

First, δ-smoothing is load-bearing. Without the floor in Eq. 8, any worst-in-set value sets $\hat{g}_i = 0$ and zeroes the entire product regardless of the other four dimensions, collapsing all such systems into an uninformative tie at the bottom. (The superseded negative-exponent formulation failed at the opposite boundary, diverging as a best-in-set cost value drove its base to zero.)

Second, we report weight sensitivity as a continuous sweep rather than as agreement among a few hand-picked profiles, for the reason set out in Section 7.2. Sweeping w_m over $[0, 0.6]$ and renormalizing the remaining weights, the harmonized-memory ranking never changes: the desktop holds rank 1 across the entire range, so its first-place finish is genuinely insensitive to how heavily memory is weighted. Under sustained-serving memory the same sweep yields a sharp crossover at $w_m \approx 0.085$, above which the laptop takes rank 1. On the configuration axis, C1 and C2 exchange rank 1 at $w_a \approx 0.075$, while C3 attains rank 1 nowhere in the sweep—its dominance failure is robust to weighting.

Third, the reference-set dependence discussed in Section 7.2 is not merely theoretical. Under harmonized memory the top rank survives dropping any single system. Under sustained-serving memory it does not: the full three-way comparison puts the laptop first, but dropping the phone restretches the min–max normalization over the two remaining, near-tied-accuracy systems and *flips* the ordering so that the desktop wins the pairwise comparison.

Fourth, and most consequentially, the measurement protocol dominates all of the above. Neither the weights nor the reference set changes the cross-tier ranking as decisively as the choice between warm-up and sustained-serving memory accounting does. Practitioners should therefore report the raw per-dimension values (Table 8) alongside any composite, state the measurement boundary and accounting protocol as part of the result, and prefer fixed anchors over set-relative normalization when longitudinal comparability matters.

Scope and limitations. This study is an illustrative proof of concept, and its claims should be read as proportional to that evidence. Five limitations bound what it supports. *(i) Point estimates.* Every configuration was run once; we report per-query means with standard deviations describing spread across queries, not confidence intervals on the means, and we make no claim of statistical significance for any comparison. Differences smaller than the per-query spread—notably the accuracy differences in Experiment 2—carry no ordering information. *(ii) Sample sizes.* The phone tier rests on 55 scored and 32 energy-instrumented queries rather than the full 200, for the reasons given above; its numbers are the weakest in the study, and the race-to-idle conclusion is stated as an ordering robust to boundary choice precisely because the margin, rather than the sample, carries it. *(iii) Measurement boundary.* Energy is not measured at a common boundary across tiers, so absolute cross-platform energy values are not directly comparable; only the ordering, bounded above, is claimed. *(iv) Memory accounting.* The desktop figure is the *host* resident set of the CUDA server process, not GPU VRAM (which WDDM does not attribute to process RSS); a deployment that restarts the server periodically or caps the host allocator would see the smaller warm-up value. *(v) No NPU.* All inference runs on CPU or CUDA GPU. **Nothing in this study is evidence about NPU behavior**, and the NPU-related discussion elsewhere in this survey rests on the cited literature, not on these measurements. Correspondingly, a 12,000-passage corpus and a 200-question set do not constitute a definitive benchmark of these devices; the contribution is the demonstration that ALEMC is measurable end-to-end and that it surfaces decision-relevant structure—protocol dependence, Pareto domination—that single-metric evaluation hides.

7.4. Case Studies

To ground the ALEMC framework in concrete deployment scenarios, we examine three application domains where on-device RAG addresses distinct operational constraints. Each case study illustrates how the four-way

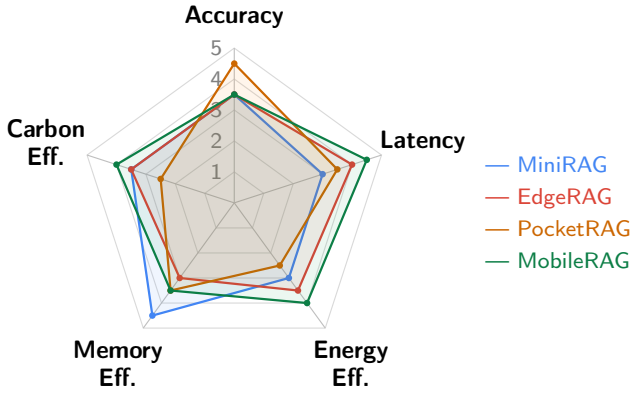


Figure 12: *Illustrative* radar chart comparing four on-device RAG systems across the five ALEMC dimensions (higher is better; latency, energy, memory, and carbon are inverted so that efficiency maps to larger radii). PocketRAG excels on accuracy due to its cross-attention reranker; MobileRAG achieves the best latency and energy efficiency through its holistic optimization stack. Scores are normalized on a 1–5 scale derived from the *reported* (and, where a dimension is unreported, estimated) metrics in [33, 107, 67, 99]; because no source reports all five dimensions, this chart is a qualitative visualization of characteristic trade-off signatures rather than a measurement. For measured ALEMC profiles on real hardware, see the empirical study in Section 7.3 (Tables 8–10).

trade-off manifests differently depending on the domain's binding constraint.

7.4.1. Healthcare: Offline First Aid and Clinical Decision Support

Healthcare represents perhaps the most compelling use case for on-device RAG, combining stringent privacy requirements with the need for offline operation in resource-limited settings.

PocketRAG [67] demonstrates a fully offline first-aid guidance system packaged as a 3.4 GB Android application. By pairing a 4-bit quantized Phi-3-mini with a hybrid BM25/dense retrieval pipeline over 15,000 medical passages, PocketRAG achieves 94.5% factual accuracy—a level sufficient for non-specialist first-aid guidance—while operating entirely without network connectivity. The system's binding constraint is *accuracy*: medical misinformation carries direct safety consequences, necessitating the use of a cross-attention reranker (12M parameters) that would be unnecessary in less safety-critical domains. This accuracy-first posture increases energy per query relative to reranker-free configurations but is justified by the application's risk profile.

Complementing this work, Medicine on the Edge [97] explores neural network pruning strategies specifically tailored for deploying medical AI on mobile hardware. The study demonstrates that structured pruning can reduce model size by 40–60% while preserving diagnostic accuracy within clinically acceptable margins, and that the pruned models' reduced computational demand translates to proportional energy savings—a critical consideration for devices operating on battery power in field clinics or disaster-response scenarios. In a similar vein, Kallakuri *et al.* [66] show that input-driven saliency adaptation can prune on-device medical assistants dynamically per query, and HealthSLM-Bench [124] provides a dedicated benchmark for small language models on mobile and wearable healthcare-monitoring tasks, giving the accuracy dimension of ALEMC a domain-appropriate reference point. Interpretability is a further salient constraint in clinical deployment: Qin *et al.* [102] propose an interactive deep survival-analysis framework with time-dependent gradient integration, illustrating how compact clinical prediction systems can expose feature-level reasoning to practitioners. This supports the thesis that on-device medical RAG should privilege architectures whose predictions can be audited locally rather than offloading reasoning to opaque cloud models.

From an ALEMC perspective, healthcare on-device RAG occupies a distinctive region of the trade-off space: accuracy is non-negotiable ($\mathcal{A} > 90\%$), memory is bounded by the deployment device ($\mathcal{M} < 4\,\text{GB}$), and energy efficiency is desirable but subordinate to correctness. Carbon considerations become salient when scaling deployment to population-level health applications in developing regions, where grid carbon intensity may be high and device charging infrastructure limited.

7.4.2. Smart Home: Always-On Ambient Intelligence

Smart home assistants present a fundamentally different constraint profile: the system must process frequent, low-complexity queries (*e.g.*, "turn off the living room lights," "what's the weather forecast?") with minimal latency and energy, while running continuously on a plugged-in device with modest computational resources.

Birkmose *et al.* [7] investigate deploying compact language models directly on smart home hubs for command interpretation and response generation. Their analysis reveals that on-device processing eliminates the 100–300 ms network round-trip to cloud services, enabling sub-second response times that significantly improve user satisfaction. Critically, local processing also avoids transmitting potentially sensitive household activity data to external servers—a growing concern as smart home devices proliferate.

Harmony [134] extends this paradigm to multi-device coordination, using a language model to interpret natural language instructions that span multiple smart home subsystems (*e.g.*, "set up movie mode"—dimming lights, closing blinds, and activating the entertainment system). The system's on-device LLM serves as a semantic router that decomposes complex instructions into device-specific API calls, with RAG providing grounding in the device documentation and user preference history.

The binding constraint for smart home RAG is *energy*: an always-on assistant processing 50–200 queries per day accumulates substantial energy consumption over its multi-year lifetime. At 10 J per query and 100 queries per day, the

annual energy consumption attributable to RAG inference alone reaches approximately 365 kJ (0.1 kWh)—modest in absolute terms but non-trivial when multiplied across millions of deployed devices. The carbon implications scale accordingly: at a global average grid intensity of approximately 450 gCO_2/kWh, 100 million smart home devices each consuming 0.1 kWh annually for RAG inference would collectively emit approximately 4,500 tonnes of CO_2 per year (an illustrative order-of-magnitude estimate—$10^8 \times 0.1\,\text{kWh} \times 450\,gCO_2/\text{kWh}$—intended to convey fleet-level scale, not a forecast for any specific deployment). This fleet-level analysis underscores that the sustainability of on-device AI is not merely a per-device concern but a systems-level challenge whose significance grows with deployment scale.

7.4.3. Finance: Secure Document Question Answering

Financial applications represent a growing use case for RAG systems where privacy and latency jointly motivate on-device deployment. Financial analysts routinely query large document corpora—10-K filings, earnings transcripts, regulatory disclosures—that contain sensitive information unsuitable for cloud transmission. That this is a hard, independently benchmarked task is established outside our own work: the FinQA [15] and TAT-QA [144] benchmarks show that question answering over financial tables and text demands multi-step numerical reasoning on which general-purpose models fall well short of expert humans, and recent systems such as the hierarchical evidence-curation retriever of Choe *et al.* [22] demonstrate that carefully structured retrieval—rather than a larger generator—is what closes the gap on open-domain financial QA over standardized documents. This is directly relevant to the resource-constrained on-device setting, where investing in retrieval quality is far cheaper than scaling the generator. Consistent with this independent evidence, our own prior work demonstrates that RAG pipelines with hybrid retrieval and neural reranking achieve 49% answer correctness on financial QA benchmarks [21], and that hybrid document-routed retrieval can further improve correctness to 67.7% by eliminating cross-document confusion [20]. For on-device deployment, these financial RAG systems face a distinctive binding constraint: the retrieval corpus is large (hundreds of multi-hundred-page documents), making compact index representations and efficient retrieval algorithms critical. The combination of scoped document routing [20] with lightweight embedding models and quantized SLMs offers a path to on-device financial RAG that preserves both data confidentiality and answer quality.

Recent evidence on financial-document retrieval sharpens this design prescription. Wu and Lin [130] show that comparing full-text sparse retrieval against a compact dense encoder is unfair when evidence lies beyond the encoder's input window; after segmenting filings into encoder-sized evidence units, combining BM25 with the compact encoder improves reference-level Hit@10 by roughly 28% over either component. Their finding that training-free reciprocal-rank fusion remains competitive with lightweight query-adaptive routers further suggests that a simple fixed fusion rule can be preferable on resource-constrained devices when the adaptive machinery adds complexity without a statistically reliable retrieval gain.

Adjacent financial decision-support studies expose complementary trade-offs that an application-specific ALEMC profile should not hide. In credit-card fraud detection, amount-conditioned training provides only small gains over conventional class weighting, whereas post-training score–amount reranking produces the largest shift in dollar recall, demonstrating that operational utility may differ from transaction-level accuracy [129]. Validation-stage combinatorial fusion can improve imbalanced fraud detection by selecting a small, complementary classifier subset rather than indiscriminately combining every model [47]; controlled encoder comparisons likewise show that no representation dominates all fraud metrics and that interpretable encodings can remain competitive with learned embeddings [46]. Finally, observation-window experiments in churn prediction find a diminishing-returns region but show that its location changes with cohort, target, and feature design [48]. Although these four studies are not RAG implementations, they provide transferable evidence for financial edge systems: accuracy should be expressed in domain-relevant utility terms, and additional models, learned representations, or historical context should be retained only when their measured benefit justifies their latency, memory, and auditability costs.

Interpretability is equally important when retrieved financial evidence informs decisions. Han *et al.* [49] combine XGBoost with TreeSHAP and ablation tests to decompose equity-return predictions into auditable factor contributions, finding that the two interpretation methods expose feature-substitution structure that either method alone can miss. Although the study is not an on-device RAG implementation, its dual-audit design provides a useful template for validating whether a compact financial RAG system's retrieved evidence and downstream attribution tell a consistent decision story.

Beyond financial question answering, domain-specific few-shot classification using LLMs has shown promise across specialized fields [84], suggesting that the RAG-plus-classification paradigm can be adapted to diverse on-device scenarios where labeled data is scarce but domain knowledge can be encoded in a retrieval corpus. Generative AI systems for simulating real-world dynamics [133] further illustrate the breadth of applications where retrieval-grounded generation can enhance the practical utility of on-device intelligence.

7.4.4. IoT Edge: Sensor-Augmented Predictive Maintenance

At the far end of the resource spectrum, IoT edge devices—industrial sensors, agricultural monitors, environmental probes—operate under extreme memory (<1 GB), compute (ARM Cortex-M/A class), and power (battery or

energy-harvesting) constraints, often with intermittent or absent network connectivity.

In predictive maintenance scenarios, RAG can augment sensor-derived anomaly signals with contextual knowledge from equipment manuals, maintenance logs, and failure mode databases. A compact SLM retrieves relevant troubleshooting procedures based on the detected anomaly pattern and generates human-readable maintenance recommendations. The retrieval corpus is small (thousands of documents) but domain-specific, and the system must function reliably in offline environments—factory floors, remote pipelines, offshore platforms—where cloud connectivity cannot be guaranteed.

The binding constraint here is *memory* and *energy* jointly. Sub-1B parameter models (MobileLLM [89] at 125–350M parameters) quantized to 4-bit precision occupy 50–200 MB, leaving room for a compact graph-based index (MiniRAG [33]) within the 512 MB–1 GB memory envelope. Energy is critical because devices may operate on battery or harvested power, requiring energy per query well below 5 J to sustain multi-month operation between charges. The carbon dimension gains a distinct interpretation in this context: IoT devices are manufactured in vast quantities (projected 75 billion by 2030), making embodied carbon per device a dominant factor in the total lifecycle assessment. Extending device lifetime through energy-efficient inference thus yields compound sustainability benefits—reducing both operational and embodied carbon by deferring hardware replacement.

7.5. Integration Patterns

Drawing on the systems reviewed in Sections 4–6 and the case studies above, we distill three recurring integration patterns that represent distinct points on the design-space Pareto frontier. Each pattern reflects a coherent set of architectural choices spanning compression, retrieval, and sustainability, and is suited to a specific deployment regime.

> ***Scope of quantitative claims (please read first).*** *Unless a number below is explicitly marked* (reported, [cite]), *it is an* (estimate): the numerical ranges in Patterns 1–3 synthesize order-of-magnitude representative values and component-level figures from the literature into coherent end-to-end profiles, sometimes combined with simple back-of-the-envelope arithmetic (for example, the weighted energy accounting in Pattern 2). They are intended as design intuition, not as a single unified benchmark or a reproducible experimental run performed for this review. For measured, end-to-end ALEMC numbers on real hardware, see the empirical study in Section 7.3.

7.5.1. Pattern 1: Quantized SLM + Lightweight Retriever + NPU

This pattern represents the *minimal viable on-device RAG* configuration, targeting fully offline, single-device deployment with no cloud dependency.

Architecture. The generator is a 1–3B parameter SLM (Phi-3-mini or Gemma-2B) quantized to 4-bit weights via AWQ [83] or GPTQ [37], with INT4/INT8 KV cache quantization per MobileQuant [115] for NPU-native execution. The retriever uses MiniRAG's [33] heterogeneous graph index, which avoids dense embedding storage and compensates for the weak semantic capabilities of small embedding models through topology-guided retrieval. Inference is dispatched to the device's NPU via QNN or Core ML, with fused operator graphs and tiled attention adapted to the NPU's SRAM geometry.

Resource profile. Model weights occupy 1.5–2.5 GB; the graph index requires 200–500 MB depending on corpus size; KV cache at INT4 precision for 2,048-token context adds approximately 150 MB. Total peak memory: 2.0–3.2 GB (estimate), fitting within the application memory budget of current mid-range smartphones. Energy per query: 2–8 J (estimate) depending on response length and NPU efficiency. TTFT: 200–500 ms (estimate).

Sustainability characteristics. Pattern 1 achieves the lowest per-query carbon footprint of the three patterns by eliminating network energy entirely and leveraging NPU energy efficiency (5–15 TOPS/W at INT8). The absence of cloud dependency also maximizes data privacy and ensures operation in connectivity-limited environments. The primary limitation is accuracy: graph-based retrieval and small generators yield lower answer quality on complex, multi-hop questions compared to patterns that incorporate reranking or cloud verification.

7.5.2. Pattern 2: Speculative Edge–Cloud Hybrid

This pattern addresses the accuracy ceiling of fully on-device RAG by introducing selective cloud verification, achieving higher answer quality while preserving local-first operation for the majority of queries.

Architecture. The local component follows Pattern 1—a quantized SLM with lightweight retrieval running entirely on-device. In addition, the system implements a *confidence-gated cloud fallback*: the local SLM generates a draft response and estimates its own confidence via calibrated token-level entropy or a lightweight learned verifier. When confidence exceeds a threshold τ, the local response is returned directly. When confidence falls below τ, the query and retrieved context are forwarded to a cloud-hosted large model (the "generalist verifier" in the Speculative RAG [128] paradigm), which either validates the local draft or generates a corrected response. This architecture extends the speculative decoding principle—a small model drafts, a large model verifies—from token-level to query-level granularity.

Resource profile. On-device resources are identical to Pattern 1. Cloud resources are consumed only for low-confidence queries; representative deployments in the literature often report fallback rates of roughly 15–30% of total query volume for well-calibrated systems (reported, [128]; the exact fraction is workload-dependent). The effective energy per query is the weighted sum: $\mathcal{E}_{\text{eff}} =$

$(1-p)\,\mathcal{E}_{\text{local}} + p\,(\mathcal{E}_{\text{local}} + \mathcal{E}_{\text{network}} + \mathcal{E}_{\text{cloud}})$, where p is the cloud fallback rate. Substituting illustrative component energies $p = 0.2$, $\mathcal{E}_{\text{local}} = 5\,\text{J}$, $\mathcal{E}_{\text{network}} = 2\,\text{J}$, and $\mathcal{E}_{\text{cloud}} = 15\,\text{J}$ (all estimates) gives $\mathcal{E}_{\text{eff}} = 0.8 \times 5 + 0.2 \times 22 = 8.4\,\text{J}$. The always-cloud baseline under the same component values is $\mathcal{E}_{\text{network}} + \mathcal{E}_{\text{cloud}} = 17\,\text{J}$, so the hybrid consumes approximately *51%* less energy per query (estimate, from this back-of-the-envelope combination).

Sustainability characteristics. Pattern 2 achieves a favorable accuracy–energy trade-off: most queries are served locally at low energy cost, while the cloud fallback catches the long tail of difficult queries that would otherwise result in incorrect or hallucinated responses. The confidence threshold τ serves as a direct control knob on the accuracy–energy Pareto frontier: lower τ reduces cloud calls (saving energy and carbon) at the cost of accepting more local errors; higher τ increases accuracy but raises average energy per query. Carbon-aware variants can dynamically adjust τ based on the cloud provider's current grid carbon intensity, preferring local responses when the cloud is carbon-expensive and relaxing the threshold when the cloud operates on renewable energy.

7.5.3. Pattern 3: Carbon-Aware Adaptive RAG

This pattern treats energy and carbon as first-class runtime optimization objectives, dynamically adapting the RAG pipeline's behavior in response to the device's energy state and the grid's carbon intensity.

Architecture. The core pipeline follows Pattern 1 or 2, but augmented with three adaptive control mechanisms. First, *DVFS-controlled inference* [86]: the NPU and CPU clock frequencies are modulated per-phase (high during compute-bound prefill, reduced during memory-bound decode), saving 20–34% energy per query without violating latency SLOs (reported, [86]). Second, *carbon-aware LoRA scaling* [142]: the system maintains multiple LoRA adapter variants at different ranks ($r \in \{4, 8, 16, 32\}$) and selects the adapter rank in real time based on the current grid carbon intensity and remaining battery level. Under high carbon intensity or low battery, the system downgrades to a lower-rank adapter (fewer parameters, less compute, lower energy) with a modest accuracy trade-off; when conditions are favorable, it upgrades to the highest-rank adapter for maximum quality. Third, *adaptive retrieval depth*: the number of retrieved passages k is dynamically adjusted based on the energy budget. Under normal conditions, $k = 5$ provides a strong accuracy–energy balance; under energy-constrained conditions (*e.g.*, battery below 20%), the system reduces to $k = 2$ and applies SRAS [96] document selection to retain only the most relevant passage, accepting a 3–5% accuracy reduction in exchange for approximately 40% energy savings (estimate).

Resource profile. Memory overhead beyond Pattern 1 is modest: storing four LoRA adapter variants at ranks 4–32 adds approximately 20–80 MB, and the SRAS policy network occupies 0.76 MB. The adaptive mechanisms introduce negligible latency overhead (<5 ms for adapter selection and retrieval depth decision). The energy profile is variable by design, ranging from 60% of Pattern 1's energy (under aggressive energy-saving mode) to 110% (under quality-maximizing mode with high-rank adapters).

Sustainability characteristics. Pattern 3 represents the fullest integration of the three pillars, operationalizing the insight that optimal system behavior is not static but context-dependent. By coupling compression, retrieval, and sustainability into a closed-loop adaptive system, Pattern 3 can reduce cumulative carbon emissions by 25–40% over a static-configuration baseline while maintaining average accuracy within 2–3% of the non-adaptive system (estimate). This pattern is particularly well-suited to always-on devices (smart home hubs, industrial monitors) where the compounding effect of per-query savings across millions of daily queries yields substantial aggregate impact.

7.6. Summary

The synthesis presented in this section demonstrates that sustainable on-device RAG is not achievable by optimizing any single pillar in isolation. Compression enables on-device deployment but determines the memory budget available for retrieval; retrieval quality directly shapes both accuracy and energy cost; and sustainability strategies modulate the system's operating point in response to real-time environmental context. The ALEMC framework provides the evaluation machinery to reason about these trade-offs systematically, while the three integration patterns offer concrete architectural templates spanning the design space from minimal-resource fully-local deployment (Pattern 1) to carbon-aware adaptive systems (Pattern 3). Collectively, these contributions aim to shift the discourse from "can we run RAG on-device?"—a question the systems in Section 5 have largely answered affirmatively—to "can we run RAG on-device *sustainably*?"—a question that requires the unified perspective this section advances.

8. Open Challenges and Future Directions

The preceding sections reveal that sustainable on-device RAG is technically feasible but far from mature. Significant gaps persist across the hardware, software, evaluation, and policy dimensions that collectively define this design space. This section identifies six open challenges, analyzes why each matters for the long-term trajectory of the field, and proposes concrete research directions for the community. Fig. 13 outlines a staged research roadmap across near-, mid-, and long-term horizons, and Table 11 provides a structured summary.

8.1. Hardware Gaps: NPUs Beyond CNNs

Current mobile NPUs were designed primarily for convolutional neural networks and fixed-size tensor operations. Their instruction set architectures (ISAs), memory hierarchies, and scheduling models reflect the regular, spatially tiled computation patterns of image classification and object

detection—workloads characterized by static shapes, dense arithmetic, and minimal control flow. Autoregressive language model inference, by contrast, demands capabilities that fall outside this design envelope.

Three architectural mismatches are particularly acute. First, *variable-length sequence handling*: RAG queries produce retrieved contexts of unpredictable length, and the KV cache grows dynamically with each generated token. NPUs that require statically shaped tensors must pad inputs to worst-case dimensions, wasting both compute and on-chip SRAM. Second, *sparse operations*: pruned models (Section 4) and sparse attention patterns yield irregular computation graphs that NPU systolic arrays cannot efficiently exploit, forcing fallback to CPU execution and negating the energy advantages of dedicated acceleration. Third, *dynamic computation graphs*: techniques such as speculative decoding, early exit, and adaptive retrieval depth produce control-flow-dependent execution paths that conflict with the ahead-of-time compilation model used by most NPU toolchains [4].

Addressing these gaps requires NPU ISA extensions specifically targeting LLM workloads: hardware scatter-gather support for sparse matrix operations, native KV cache management instructions that handle dynamic allocation and eviction without CPU intervention, and variable-length tiling modes that adapt on-chip buffer partitioning to the actual sequence length at runtime. The eIQ Neutron compiler [4] represents an early step toward NPU-aware compilation for language models, but its optimizations remain limited to static-shape inference. AutoNeural [13] demonstrates that architecture-level co-design—replacing attention-heavy components with convolution-based alternatives that align with NPU operator sets—can yield order-of-magnitude latency improvements on current NPU hardware without waiting for ISA extensions. A broader hardware–software co-design effort—analogous to how GPU architectures evolved tensor cores to accelerate matrix multiplications—is needed to make NPUs first-class citizens for autoregressive generation and RAG retrieval.

8.2. Standardized Benchmarks for On-Device RAG Energy

The on-device RAG literature currently lacks a standardized benchmark for energy measurement. Results reported across papers use different hardware platforms, power measurement methodologies (software estimation vs. hardware power monitors), workload definitions (varying query distributions, corpus sizes, and context lengths), and environmental conditions (thermal throttling, battery state, background processes). Consequently, energy and latency figures from different studies are largely incomparable, making it impossible to draw reliable conclusions about which system designs are most efficient.

We propose extending the MLPerf Inference benchmark suite [103] with RAG-specific scenarios. MLPerf already defines standardized workloads, measurement protocols, and reporting formats for neural network inference across datacenter and edge settings; adding RAG scenarios would leverage this existing infrastructure while addressing several gaps. Specifically, a RAG benchmark should define: (i) a standardized document corpus and query set with known ground-truth answers, enabling joint measurement of accuracy and energy; (ii) a prescribed measurement protocol specifying the power instrumentation (external power analyzer vs. on-chip sensors), warm-up procedure, and statistical reporting requirements (median, tail latency, confidence intervals); (iii) workload profiles that separately stress the retrieval and generation phases, since their computational characteristics differ fundamentally (Section 6.1); and (iv) a device taxonomy that enables fair comparison across NPU generations and vendors.

Until such a benchmark exists, researchers should at minimum report energy measurements using calibrated external power analyzers, specify the exact hardware platform and software stack, disclose the power measurement methodology and averaging window, and provide per-phase (retrieval vs. generation) breakdowns. These practices would substantially improve the reproducibility and comparability of on-device RAG energy results, even in the absence of a formal benchmark.

8.3. Lifecycle-Aware Model Selection

Current model selection practices focus almost exclusively on inference-time metrics—accuracy, latency, memory footprint, and increasingly, energy per query. This perspective ignores the substantial carbon emissions incurred during model training, which for large language models can reach hundreds of tonnes of CO_2e [111]. A lifecycle-aware perspective reframes the question: when the training carbon is amortized over the model's total deployment lifetime, which model truly minimizes per-query carbon?

The answer depends critically on deployment scale and retraining frequency. Consider two scenarios: a 7B-parameter model trained once at high carbon cost and deployed across 10 million devices for two years, versus a 1B-parameter model that requires monthly retraining to remain current. If the larger model's one-time training cost of 50 tCO_2e (5×10^7 g) is amortized over $10^7\times365\times2\times20 \approx 1.46 \times 10^{11}$ queries (10 million devices, two years, 20 queries per device per day), the per-query training carbon is negligible ($\sim3.4\times10^{-4}$ gCO_2e). The smaller model's monthly retraining, even at 2 tCO_2e per cycle, accumulates 48 tCO_2e over two years—a comparable total despite the dramatically lower per-cycle cost. When combined with the larger model's potentially lower per-query inference energy (due to fewer retrieval augmentation steps needed to compensate for weaker parametric knowledge), the lifecycle calculus may favor the seemingly "less efficient" larger model.

The research community needs *carbon lifecycle calculators* that integrate training emissions, deployment scale, retraining cadence, inference energy, and grid carbon intensity into a unified decision framework. Such tools would extend the ALEMC evaluation framework proposed in Section 7 with a temporal dimension, enabling practitioners to project

total lifecycle carbon under different deployment scenarios. CO2-Meter [40] has taken initial steps by incorporating embodied carbon into per-query estimates, but a comprehensive lifecycle model that spans training through end-of-life device disposal remains an open research problem. Recent work further argues that lifecycle analysis must also account for upstream material supply chains: He *et al.* [51] show that rare-earth bottlenecks introduce cost and schedule risks for large AI data centers that are not captured by carbon-only accounting, strengthening the case for edge-first designs that amortize specialized hardware over many low-energy queries rather than concentrating demand in centralized GPU clusters.

8.4. Federated and Privacy-Preserving RAG

On-device RAG inherently preserves user privacy by keeping both queries and retrieved documents on the local device, eliminating the data transmission that characterizes cloud-based alternatives (Section 6.3). However, this privacy advantage comes at a cost: each device's knowledge base is limited to its locally stored corpus, preventing the system from benefiting from collective knowledge across a user population.

Federated RAG architectures address this tension by enabling collaborative knowledge sharing without centralizing sensitive data. Chakraborty *et al.* [11] propose a federated retrieval-augmented generation framework in which devices share retrieval index statistics—cluster centroids, embedding distribution summaries, and relevance feedback signals—without transmitting raw documents or user queries. A central aggregator synthesizes these statistics into a global retrieval model that improves retrieval quality for all participants, while each device retains full control over its local corpus.

Several technical challenges must be resolved before federated RAG becomes practical. *Secure aggregation of retrieval indices* is non-trivial: cluster centroids and embedding statistics can leak information about the underlying documents, particularly when the corpus is small or domain-specific. Cryptographic protocols such as secure multi-party computation can provide formal privacy guarantees but introduce substantial computational overhead that may be prohibitive on resource-constrained devices. *Differential privacy for embeddings* offers a more lightweight alternative, adding calibrated noise to shared embedding representations to bound the information leakage per document. However, the accuracy–privacy trade-off for dense retrieval embeddings is poorly characterized: the noise required for meaningful privacy guarantees may degrade retrieval recall below acceptable thresholds.

Additionally, the *communication cost* of federated RAG must be carefully managed. Transmitting embedding updates, even in compressed form, consumes network bandwidth and energy—partially offsetting the privacy and energy advantages of local processing. Designing communication-efficient federated protocols that achieve convergence with minimal round-trips is an active research frontier that intersects with both the federated learning and information retrieval communities.

As on-device RAG systems grow toward multi-session agentic operation—a direction surveyed by Sharma *et al.* [109], who argue that small language models are well matched to the bounded, tool-mediated subtasks that agentic pipelines decompose into—lightweight memory architectures that explicitly model semantic, temporal, and causal relations—such as MAGMA [58]—will need to be co-designed with NPU-aware retrieval and energy budgets, an integration gap further confirmed by empirical analysis showing that current Memory-Augmented Generation systems consistently fall short of their theoretical performance in long-horizon tasks [59].

Privacy concerns also extend beyond data transmission to the model's own behavior: on-device RAG systems that process personal documents remain susceptible to output-side PII leakage [127] and, when operating agentically, to prompt injection and adversarial-retrieval attacks [60]—motivating privacy-preserving inference mechanisms (output filtering, differential privacy in generation) as integral rather than optional components of the design.

Model unlearning adds a further audit challenge when local assistants expose reasoning traces. Li *et al.* [76] show that a gap between apparently unlearned answers and traces that still emit forgotten content does not by itself identify weight-level memorization; decode-time template swaps can materially change the apparent bypass signal. On-device unlearning evaluations should therefore audit both answers and traces while controlling the decoding template, rather than interpreting a parser-split gap as conclusive evidence that private content remains memorized.

8.5. Multi-Modal On-Device RAG

Current on-device RAG systems operate almost exclusively over text corpora. However, many real-world applications demand retrieval across heterogeneous modalities: a healthcare assistant may need to retrieve from medical images, clinical notes, and sensor telemetry simultaneously; a field maintenance system may index equipment photographs, technical manuals, and audio recordings of machine sounds. The emerging generation of multi-modal small language models—vision-language models such as Gemma with visual adapters, LLaVA variants below 3B parameters, and audio-conditioned SLMs—opens the possibility of multi-modal RAG on edge devices.

The technical challenges are substantial. Multi-modal embeddings require separate or unified encoder architectures that map images, text, and audio into a shared retrieval space. On-device deployment of even a lightweight vision encoder (*e.g.*, a quantized ViT-Small at ∼200 MB) alongside a text embedding model and a generative SLM pushes memory budgets beyond current limits for sub-8 GB devices. The retrieval index must accommodate heterogeneous embedding dimensions and similarity metrics, complicating the vector search infrastructure discussed in Section 5. Most critically, the energy cost of multi-modal inference scales with the

number of modalities processed per query: encoding an image consumes 2–5× more energy than encoding a text passage of equivalent information content, and multi-modal attention mechanisms that fuse cross-modal representations add further computational overhead.

Efficient multi-modal retrieval on edge devices will likely require modality-specific compression strategies (*e.g.*, aggressive quantization for visual encoders, which tolerate precision reduction better than language models), modality-aware retrieval routing that invokes expensive encoders only when the query demands it, and hardware support for heterogeneous inference workloads that can dynamically allocate NPU resources across modalities. AutoNeural [13] provides early evidence that NPU-native co-design can address the quantization and memory challenges of multi-modal models, achieving 14× latency reduction for vision-language inference through architecture-level redesign.

For retrieval specifically, FineCIR [79] parses fine-grained modification semantics for composed image retrieval, while ReTrack [80] uses evidence-driven dual streams for composed video retrieval. A broader family of composed-retrieval methods supplies additional building blocks for future edge systems. ENCODER mines visual entities and binds them to textual modification actions [78], whereas OFFSET combines dominant-region segmentation with text-guided visual-focus revision [17]; both illustrate selective cross-modal processing that could reduce irrelevant features before indexing. Robustness to imperfect multimodal data is addressed by HABIT's progressive identification of clean image–text triplets [81] and INTENT's joint treatment of cross-modal correspondence noise and modality-inherent visual noise [16]. For video, HUD uses hierarchical uncertainty modeling to resolve referential ambiguity and align fine-grained semantics [18], while REFINE separates shared and frame-differential semantics before modification-guided fusion [55]. These mechanisms could inform compact cross-modal indices and selective fusion policies, although none of these composed-retrieval studies yet measures latency, memory, or energy on edge hardware.

Advances in medical imaging feature extraction [137] further motivate the multi-modal edge RAG agenda, as clinical decision support systems increasingly require joint reasoning over imaging data and textual records under strict privacy and latency constraints. This research direction sits at the intersection of multi-modal learning, efficient inference, and edge systems—a convergence that mirrors the three-pillar structure of this survey at a higher level of complexity.

Financial applications provide another motivation for heterogeneous retrieval. Dai *et al.* [24] show that satellite imagery supplies timely firm-level signals that reduce information asymmetry and help explain Chinese B-share discounts, demonstrating that economically material evidence can reside outside conventional filings and news. A privacy-preserving financial assistant could therefore retrieve compact visual indicators alongside local text documents, provided that modality routing prevents expensive vision encoding on text-only queries.

8.6. Policy and Regulation

The environmental impact of AI is increasingly entering the regulatory discourse. The European Union's AI Act, which entered into force in 2024, establishes risk-based requirements for AI systems deployed in the EU, including transparency obligations and conformity assessments. While the current text does not mandate energy or carbon reporting, several provisions create regulatory hooks that could evolve in this direction: high-risk AI systems must document their "resource requirements" and "energy consumption," and the Act's implementing legislation is expected to specify reporting granularity over the coming years. Similar regulatory efforts are emerging in the United States (Executive Order on AI, NIST AI Risk Management Framework), China (Interim Measures for Generative AI), and at international bodies including the OECD and the G7 Hiroshima AI Process.

Schwartz *et al.* [106] were among the first to advocate for mandatory efficiency reporting in AI research, proposing that publications should disclose computational cost alongside accuracy metrics. Extending this principle to deployed systems suggests a model of *carbon labeling for AI*—analogous to nutritional labels on food products or energy efficiency ratings on appliances. A carbon label for an AI model or service might report: training carbon (total and amortized per query), inference energy per query at specified hardware and batch configurations, and the carbon intensity assumptions underlying the reported figures.

For on-device AI specifically, regulation could drive several beneficial outcomes. Mandatory energy reporting would create market incentives for efficient model design and NPU optimization, as consumers and procurement officers could compare the energy cost of competing AI-enabled devices. Standardized carbon labels would require the measurement infrastructure and benchmarks discussed in Section 8.2, accelerating the development of tools that the research community already needs. However, regulation also carries risks: premature standardization could lock in measurement methodologies that fail to capture the full lifecycle, and compliance costs could disproportionately burden smaller developers and open-source projects that lack the resources for formal carbon auditing.

The research community can proactively shape this regulatory landscape by developing robust, open measurement standards; publishing energy and carbon data alongside accuracy results (as advocated by the Green AI agenda); and engaging with policymakers to ensure that regulations are technically grounded and incentivize genuine efficiency improvements rather than superficial reporting compliance. Fairness considerations further complicate the regulatory picture: Guo *et al.* [45] survey the origins, evaluation, and mitigation of bias in LLMs

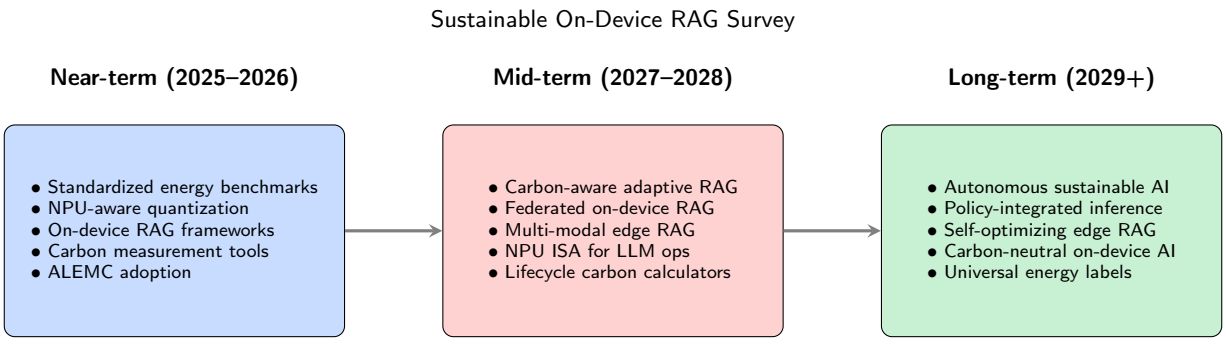


Figure 13: Research roadmap for sustainable on-device RAG across near-, mid-, and long-term horizons.

and show that model compression and fine-tuning—techniques central to on-device deployment—can amplify social biases in unpredictable ways. Temporal aggregation methods for detecting abnormal feedback patterns [100] offer a complementary monitoring signal for identifying coordinated or anomalous user responses after deployment. Any sustainability-oriented labeling scheme for on-device RAG will therefore need to co-report bias and feedback-integrity metrics alongside energy and carbon, lest efficiency gains be achieved at the cost of equitable or reliable service.

Reporting rules must also address *greenwashing*: sustainability claims can be formally disclosed yet still overstate environmental performance. Evidence that analyst scrutiny can reduce corporate greenwashing by lowering information asymmetry and strengthening external monitoring [88] supports independent verification of AI energy and carbon claims rather than reliance on self-reported labels alone. For on-device RAG, reproducible measurement logs and auditable lifecycle assumptions should therefore accompany any "green" deployment claim.

Adjacent work also illustrates how compact specialization mechanisms could reduce repeated full-model retraining. FineEdit [136] targets precise, localized text modifications, while FlowMalTrans [54] uses a flow-adapter architecture for domain-specific binary-code translation. Although these studies do not evaluate sustainable on-device RAG directly, their modular editing and adapter designs motivate future work on lightweight, updateable domain modules that can be distributed without replacing an entire edge model.

9. Conclusion

This survey reviewed 142 papers at the three-way intersection of on-device AI inference, retrieval-augmented generation, and Green AI, providing what is, to our knowledge, the first unified synthesis of a design space that prior single- and dual-pillar surveys have addressed only in fragments. By tracing the evolution from large-scale cloud language models to small, quantized models deployed on NPU-equipped edge devices, and by examining how RAG pipelines must be restructured to operate within the memory, latency, and energy constraints of these platforms, we have established that sustainable on-device RAG is both technically achievable and scientifically underexplored.

Three findings emerge from this synthesis. First, Small Language Models combined with 4-bit quantization and lightweight retrieval components can deliver practical on-device RAG performance—answering knowledge-grounded queries in under two seconds on current mobile hardware with memory footprints below 4 GB. Second, the energy and carbon costs of AI inference are measurable at per-query granularity using tools such as CodeCarbon, CO2-Meter, and EnviroLLM, yet there is no standardized protocol for reporting these costs across on-device RAG configurations, making reproducible comparison across studies difficult. Third, the ALEMC framework proposed in this survey—extending conventional accuracy, latency, and memory metrics with explicit Energy and Carbon dimensions—provides a principled path toward holistic evaluation that accounts for sustainability alongside task performance.

The novelty of this survey lies not in any single pillar but in the synthesis: by unifying on-device AI, RAG, and sustainability into a coherent analytical framework, we expose interdependencies—between quantization bit-width and energy per token, between retrieval strategy and computational overhead, between deployment location and the balance of operational versus embodied carbon—that remain invisible to surveys confined to a single domain.

Looking forward, we call on the research community to prioritize three areas of effort. First, *standardized energy and carbon benchmarks* for on-device RAG are essential; without agreed-upon measurement protocols, efficiency claims will remain fragmented and incomparable. Second, *NPU-aware RAG system design* must move beyond treating the neural accelerator as a black-box inference engine; co-designing retrieval, context compression, and generation stages around NPU datapath constraints can unlock substantial efficiency gains. Third, *carbon-conscious model selection*—choosing not only the most accurate model but the one whose lifecycle carbon cost is justified by its deployment context—should become a standard practice in on-device AI engineering.

As NPU capabilities scale and regulatory pressure on AI energy intensifies, the convergence examined here will shift from academic curiosity to engineering imperative. Sustainable on-device intelligence is no longer aspirational; it is the next necessary step.

Table 11
Summary of Open Challenges and Future Directions for Sustainable On-Device RAG

Challenge	Current State	Key Gap	Proposed Direction	Timeline
Hardware gaps (Sec. 8.1)	NPUs optimized for CNNs and fixed-size tensors	No native support for variable-length sequences, sparse ops, or dynamic graphs in LLM/RAG workloads	NPU ISA extensions for KV cache management, sparse matrix ops, and variable-length tiling	Mid-term
Standardized benchmarks (Sec. 8.2)	No reproducible energy benchmark for on-device RAG; incomparable results across papers	Missing standardized workloads, measurement protocols, and device taxonomy	Extend MLPerf with RAG-specific scenarios; define per-phase energy measurement protocols	Near-term
Lifecycle-aware selection (Sec. 8.3)	Model selection based solely on inference-time metrics	Training carbon not amortized over deployment lifetime; retraining costs ignored	Carbon lifecycle calculators integrating training, deployment scale, and retraining cadence	Near-term
Federated RAG & agentic memory (Sec. 8.4)	On-device RAG limited to local corpora; agentic memory architectures [58, 59] not yet co-designed with NPU or energy budgets	Secure index aggregation, differential privacy for embeddings, and semantic/temporal/causal memory integration unresolved	Federated retrieval protocols with privacy-preserving index sharing; NPU-aware agentic memory co-design	Mid-term
Multi-modal RAG (Sec. 8.5)	Text-only on-device RAG; multi-modal SLMs emerging but energy-intensive	No efficient framework for cross-modal retrieval on edge devices	Modality-aware compression, retrieval routing, and heterogeneous NPU scheduling	Long-term
Policy and regulation (Sec. 8.6)	EU AI Act mentions resource requirements; no mandatory energy/carbon reporting yet	No standardized carbon labeling for AI models or services	Carbon labels for AI (training + inference); open measurement standards to support regulation	Mid-term

Table 12
Software, model, and instrumentation manifest for the Section 7.3 empirical study.

Component	Version / setting
Generator	Gemma-2-2B-it, 4-bit `Q4_K_M`
Inference runtime	`llama.cpp` [43], commit `6658925`
Query embedder	`bge-small-en-v1.5` (ONNX)
Embedding runtime	`fastembed` 0.8.0 on `onnxruntime` 1.27.0
Vector search	FAISS [64] 1.13.0, `IndexFlatIP`
Retrieval depth	$k = 5$ (baseline; varied in Exp. 2)
Corpus	12,000 Wikipedia-derived passages
Query set	200 questions (100 SQuAD, 100 HotpotQA)
Decoding	greedy, temperature 0, 256-token cap, fixed prompt
Warm-up	5 queries, excluded from all statistics
Energy (Snapdragon)	whole-device (WMI battery; `adb` fuel-gauge)
Energy (RTX)	GPU-package via NVML
Memory	host RSS: warm-up (harmonized) and sustained peak

Data Availability Statement

The code and data supporting the empirical study in Section 7.3 are openly available at `https://github.com/zhycheng614/alemc-ondevice-rag` (source code under the MIT license; derived data under CC BY-SA 4.0), with an archived, version-of-record release deposited at Zenodo (`https://doi.org/10.5281/zenodo.21541720`). The repository contains the retrieval pipeline, per-device energy and memory instrumentation, the frozen evaluation queries, the 12,000-passage corpus and prebuilt index, all raw per-query logs, and the analysis scripts that generate the reported tables and figures; the software manifest is summarized in the Reproducibility Statement (Table 12). Model weights are not redistributed and are obtained separately as described in the repository.

Reproducibility Statement

Table 12 consolidates the exact software versions, decoding parameters, and instrumentation used in the empirical study of Section 7.3, complementing the inline description in the setup paragraph. The complete manifest, the frozen 200-question query set, the 12,000-passage corpus and prebuilt `IndexFlatIP` index, all raw per-query logs, and the analysis scripts that regenerate every reported table and figure are released in the repository cited in the Data Availability Statement.

A. Master Literature Classification

Table 13 maps each of the 142 reviewed papers to its category, publication year, and venue. Categories correspond to the taxonomy defined in Section 2: **A** Foundation, **B** Small Language Models, **C** Quantization & Compression, **D** NPU & Hardware, **E** RAG Core, **F** On-Device RAG, **G** Green AI, **H** Evaluation, **I** Applications.

Table 13: Master Literature Classification: All 142 Reviewed Papers by Category, Year, and Venue

Cat.	Reference	Short Title	Year	Venue
A. Foundation & Background (8 papers)				
A	Vaswani *et al.* [118]	Attention Is All You Need	2017	NeurIPS
A	Devlin *et al.* [30]	BERT	2019	NAACL-HLT
A	Brown *et al.* [8]	GPT-3	2020	NeurIPS
A	Kaplan *et al.* [68]	Scaling Laws for Neural LMs	2020	arXiv
A	Touvron *et al.* [117]	LLaMA	2023	arXiv
A	Tay *et al.* [116]	Efficient Transformers Survey	2022	ACM CSUR
A	Zhao *et al.* [141]	Survey of LLMs	2023	arXiv
A	Kaddour *et al.* [65]	Challenges & Applications of LLMs	2023	arXiv
B. Small Language Models (14 papers)				
B	Liu *et al.* [89]	MobileLLM	2024	ICML
B	Zhang *et al.* [140]	TinyLlama	2024	arXiv
B	Gunasekar *et al.* [44]	Phi (Textbooks Are All You Need)	2023	arXiv
B	Gemma Team [42]	Gemma	2024	arXiv
B	Sun *et al.* [114]	MobileBERT	2020	ACL
B	Jiang *et al.* [57]	Mistral 7B	2023	arXiv
B	Sanh *et al.* [105]	DistilBERT	2019	EMC^2 @ NeurIPS
B	Wang *et al.* [121]	Comprehensive Survey of SLMs	2024	arXiv
B	Sharma *et al.* [109]	SLMs for Agentic Systems	2025	arXiv
B	Bini *et al.* [6]	MemLoRA	2025	arXiv
B	Wang *et al.* [123]	MiniLM	2020	NeurIPS
B	Cao *et al.* [10]	Task-Specific SLM Efficiency	2026	arXiv
B	Hu *et al.* [53]	CoDES (Context-Efficient SLMs)	2026	Preprints.org
B	Zeng *et al.* [136]	FineEdit (LLM Text Editing)	2025	EMNLP Findings
C. Quantization & Compression (23 papers)				
C	Dettmers *et al.* [27]	LLM.int8()	2022	NeurIPS
C	Xiao *et al.* [131]	SmoothQuant	2023	ICML
C	Wang *et al.* [122]	BitNet	2023	arXiv
C	Ma *et al.* [93]	BitNet b1.58	2024	arXiv
C	Frantar *et al.* [37]	GPTQ	2023	ICLR
C	Lin *et al.* [83]	AWQ	2024	MLSys
C	Dettmers *et al.* [28]	QLoRA	2023	NeurIPS
C	Alizadeh *et al.* [2]	LLM in a Flash	2024	ACL
C	Frantar *et al.* [36]	SparseGPT	2023	ICML
C	Zhu *et al.* [145]	Survey on Model Compression	2024	TACL
C	Hu *et al.* [52]	LoRA	2022	ICLR
C	Dao *et al.* [26]	FlashAttention	2022	NeurIPS
C	Dao [25]	FlashAttention-2	2024	ICLR
C	Kim *et al.* [70]	FlexiQ	2025	arXiv
C	Tan *et al.* [115]	MobileQuant	2024	EMNLP Findings
C	Lu *et al.* [91]	Reassessing Layer Pruning in LLMs	2026	ICLR
C	Lu *et al.* [92]	Generic Layer Pruning (Signal)	2024	IEEE TCCN
C	Lu *et al.* [90]	Dataset Distillation (Contrastive)	2025	Neurocomputing
C	Li *et al.* [75]	SEC (Data Efficiency + Compression)	2024	IEEE EIECS
C	Jiang *et al.* [61]	DRP (Distilled Reasoning Pruning)	2026	arXiv
C	Zhang *et al.* [138]	Router-Guided Multi-Teacher Distillation	2026	arXiv
C	Jiang *et al.* [62]	Rock Tokens in On-Policy Distillation	2026	arXiv
C	Hu *et al.* [54]	FlowMalTrans (Flow-Adapter)	2025	arXiv
D. NPU & Hardware (8 papers)				
D	Xu *et al.* [132]	On-Device LMs: Comprehensive Review	2024	arXiv
D	Zheng *et al.* [143]	Edge LLMs Review	2025	ACM CSUR
D	Reddi *et al.* [103]	MLPerf Inference Benchmark	2020	ISCA
D	Bamberg *et al.* [4]	eIQ Neutron NPU Compiler	2025	arXiv
D	Ignatov *et al.* [56]	AI Benchmark (Android)	2019	ECCV-W
D	Wang *et al.* [125]	Empowering Edge Intelligence	2025	ACM CSUR
D	Chen *et al.* [12]	On-Device LLMs: Compression & Systems	2024	Preprint
D	Chen *et al.* [13]	AutoNeural (NPU VLM Co-design)	2025	arXiv
E. RAG Core (20 papers)				
E	Lewis *et al.* [73]	RAG for Knowledge-Intensive NLP	2020	NeurIPS
E	Karpukhin *et al.* [69]	Dense Passage Retrieval (DPR)	2020	EMNLP
E	Johnson *et al.* [64]	FAISS: Billion-scale Similarity Search	2021	IEEE TBDATA
E	Liu *et al.* [85]	Lost in the Middle	2024	TACL
E	Gao *et al.* [41]	RAG Survey	2024	arXiv
E	Sharma [108]	Comprehensive RAG Survey	2025	arXiv
E	Barnett *et al.* [5]	Seven Failure Points of RAG	2024	IEEE CAIN
E	Mallen *et al.* [94]	When Not to Trust LMs	2024	ACL
E	Jin *et al.* [63]	FlashRAG Modular Toolkit	2025	WWW
E	Cheng *et al.* [21]	Financial RAG with Reranking	2026	arXiv
E	Cheng *et al.* [20]	Hybrid Document-Routed Retrieval	2026	arXiv
E	Lai *et al.* [72]	Embedding Methods for Short Texts	2026	Research Square
E	Li *et al.* [79]	FineCIR (Composed Image Retrieval)	2025	arXiv
E	Wu and Lin [130]	Evidence-Unit Fairness for Financial Retrieval	2026	arXiv
E	Li *et al.* [81]	HABIT (Robust Composed Image Retrieval)	2026	AAAI
E	Li *et al.* [78]	ENCODER (Entity–Action Binding)	2025	AAAI
E	Chen *et al.* [17]	OFFSET (Focus-Shift Revision)	2025	ACM MM
E	Chen *et al.* [16]	INTENT (Noise-Robust Image Retrieval)	2026	AAAI
E	Chen *et al.* [14]	Context Compliance under Knowledge Conflict	2026	arXiv
E	Qian *et al.* [101]	Evidence-Force Calibration for Cited RAG	2026	arXiv
F. On-Device / Edge RAG (19 papers)				
F	Fan *et al.* [33]	MiniRAG	2025	arXiv
F	Seemakhupt *et al.* [107]	EdgeRAG	2024	arXiv
F	Zhu *et al.* [146]	SparseRAG	2025	ICLR

(continued on next page)

Table 13: Master Literature Classification (continued)

Cat.	Reference	Short Title	Year	Venue
F	Wang *et al.* [128]	Speculative RAG	2025	ICLR
F	Kang *et al.* [67]	PocketRAG	2026	arXiv
F	Park *et al.* [99]	MobileRAG	2025	arXiv
F	Muttur [96]	SRAS (RL Document Selector)	2025	arXiv
F	Chakraborty *et al.* [11]	Federated RAG	2025	EMNLP Findings
F	Fang *et al.* [34]	AttentionRAG (Context Pruning)	2025	arXiv
F	Shi *et al.* [110]	Reasoning in Trees (Multi-hop RAG)	2026	arXiv
F	Wang *et al.* [126]	SWE-Pruner (Context Pruning)	2026	arXiv
F	Su *et al.* [112]	Difficulty-Aware Agentic Orchestration	2026	WWW
F	Cai *et al.* [9]	Tone Sensitivity in LLMs	2025	arXiv
F	Li *et al.* [80]	ReTrack (Video Retrieval)	2026	AAAI
F	Jiang *et al.* [58]	MAGMA (Multi-Graph Agentic Memory)	2026	arXiv
F	Jiang *et al.* [59]	Anatomy of Agentic Memory (Survey)	2026	arXiv
F	Chen *et al.* [18]	HUD (Uncertainty-Aware Video Retrieval)	2025	ACM MM
F	Hu *et al.* [55]	REFINE (Composed Video Retrieval)	2026	ACM TOMM
F	Cheng *et al.* [19]	Energy-Efficient On-Device RAG on a Mobile NPU	2026	arXiv
G. Green AI & Sustainability (19 papers)				
G	Strubell *et al.* [111]	Energy and Policy for Deep Learning	2019	ACL
G	Verdecchia *et al.* [119]	Systematic Review of Green AI	2023	WIREs DMKD
G	Schwartz *et al.* [106]	Green AI	2020	CACM
G	Fu *et al.* [40]	CO2-Meter	2025	arXiv
G	Allen [3]	EnviroLLM	2025	arXiv
G	Liu *et al.* [86]	GreenLLM	2025	arXiv
G	Courty *et al.* [23]	CodeCarbon	2023	Software
G	You *et al.* [135]	Zeus	2023	NSDI
G	Dodge *et al.* [31]	Measuring Carbon Intensity of AI	2022	FAccT
G	de Vries [120]	Growing Energy Footprint of AI	2023	Joule
G	AI-CARE Team [1]	AI-CARE	2026	arXiv
G	Zheng *et al.* [142]	Carbon-Aware LoRA Scaling	2025	NeurIPS-W
G	Rojahn *et al.* [104]	Green AI Meta-analysis	2025	arXiv
G	Devadas *et al.* [29]	Carbon-Aware AI PRISMA Review	2026	Energy Informatics
G	Lacoste *et al.* [71]	Quantifying Carbon Emissions of ML	2019	arXiv
G	Fu *et al.* [39]	GaiaFlow (Carbon-Frugal Search)	2026	arXiv
G	He *et al.* [51]	Rare-Earth Bottlenecks in AI Data Centers	2026	SSRN
G	Zhang *et al.* [139]	Carbon-Aware Compute–Power Scheduling	2026	IEEE RTSI
G	Liu *et al.* [88]	Analyst Monitoring of Greenwashing	2023	Finance Res. Lett.
H. Evaluation & Benchmarks (14 papers)				
H	Es *et al.* [32]	RAGAS	2024	EACL
H	Liang *et al.* [82]	HELM	2023	Ann. NYAS
H	Muennighoff *et al.* [95]	MTEB	2023	EACL
H	Liu *et al.* [87]	Efficient Inference Survey	2025	arXiv
H	Fu *et al.* [38]	NeuroPareto (Multi-Objective Opt.)	2026	arXiv
H	Feng *et al.* [35]	Noisy but Valid (Robust LLM Eval)	2026	ICLR
H	Qi *et al.* [100]	Temporal Feedback Patterns	2026	arXiv
H	He [50]	Unified Metric Architecture for AI Infra	2025	arXiv
H	Guo *et al.* [45]	Bias in LLMs: Origin, Eval., Mitigation	2024	arXiv
H	Zhuang *et al.* [147]	Paired-MDE Budget for 4-bit Benchmarks	2026	arXiv
H	Li *et al.* [77]	SafetyRepro (Rank Instability)	2026	arXiv
H	Sun *et al.* [113]	Bias Acknowledgment in Reasoning	2026	ICML-W
H	Li *et al.* [76]	Reasoning-Trace Memorization Audit	2026	arXiv
H	Li *et al.* [74]	LLM-as-a-Judge Survey	2025	EMNLP
I. Applications & Case Studies (17 papers)				
I	Birkmose *et al.* [7]	On-Device LLM for Smart Home	2025	W-NUT
I	Yin *et al.* [134]	Harmony (Smart Home LLM)	2024	arXiv
I	Nissen *et al.* [97]	Medicine on the Edge	2025	arXiv
I	Wang *et al.* [124]	HealthSLM-Bench	2025	arXiv
I	Kallakuri *et al.* [66]	On-Device Medical AI (Pruning)	2025	arXiv
I	Liu *et al.* [84]	PoliPrompt (Domain-Specific LLM)	2024	arXiv
I	Yao *et al.* [133]	Generative AI for Real-World Dynamics	2025	TechRxiv
I	Zhan *et al.* [137]	Medical Imaging Feature Extraction	2024	arXiv
I	Wang *et al.* [127]	VisualLeakBench (Privacy/Security)	2026	arXiv
I	Jiang *et al.* [60]	Agentic AI Cybersecurity Risks	2026	arXiv
I	Qin *et al.* [102]	Interpretable Deep Survival Analysis	2025	IEEE ICDM
I	Wu [129]	Dollar-Metric Credit-Card Fraud Detection	2026	arXiv
I	Han and Wu [47]	Combinatorial Fusion for Fraud Detection	2026	arXiv
I	Han *et al.* [46]	Encoders for High-Cardinality Fraud Detection	2026	arXiv
I	Han *et al.* [48]	Observation-Window Sufficiency for Churn	2026	arXiv
I	Han *et al.* [49]	Interpretable Factor Decomposition for Markets	2026	arXiv
I	Dai *et al.* [24]	Satellite Imagery and B-Share Discount	2023	China Econ. Rev.

CRediT authorship contribution statement

Zhiyuan Cheng: Conceptualization, Methodology, Investigation, Formal analysis, Writing – Original draft, Project administration. **Longying Lai:** Investigation, Data curation, Formal analysis, Visualization, Writing – Review & Editing. **Yue Liu:** Formal analysis, Writing – Review & Editing. **Yu Sun:** Formal analysis, Writing – Review & Editing.

References

[1] AI-CARE Team, 2026. AI-CARE: Carbon-aware reporting evaluation metric for AI models. ArXiv:2602.16042.

[2] Alizadeh, K., Mirzadeh, S.I., Belenko, D., Khatamifard, S.K., Cho, M., Del Mundo, C.C., Rastegari, M., Farajtabar, M., 2024. LLM in a flash: Efficient large language model inference with limited memory, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Bangkok, Thailand. pp. 12562–12584. doi:10.18653/v1/2024.acl-long.678. also arXiv:2312.11514.

[3] Allen, T., 2025. EnviroLLM: Resource tracking and optimization for local AI. ArXiv:2512.12004.

[4] Bamberg, L., Minnella, F., Bosio, R., Ottati, F., Wang, Y., Lee, J., Lavagno, L., Fuks, A., 2025. eIQ neutron: Redefining edge-AI inference with integrated NPU and compiler innovations. ArXiv:2509.14388.

[5] Barnett, S., Kurniawan, S., Thudumu, S., Brannelly, Z., Abdelrazek, M., 2024. Seven failure points when engineering a retrieval augmented generation system, in: Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering (CAIN).

[6] Bini, M., Bohdal, O., Michieli, U., Akata, Z., Ozay, M., Ceritli, T., 2025. MemLoRA: Distilling expert adapters for on-device memory systems. ArXiv:2512.04763.

[7] Birkmose, R., Reece, N.M., Norvin, E.H., Bjerva, J., Zhang, M., 2025. On-device LLMs for home assistant: Dual role in intent detection and response generation, in: Proceedings of the Tenth Workshop on Noisy and User-generated Text (W-NUT 2025). ArXiv:2502.12923.

[8] Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., 2020. Language models are few-shot learners, in: Advances in Neural Information Processing Systems, pp. 1877–1901.

[9] Cai, H., Shen, B., Jin, L., Hu, L., Fan, X., 2025. Does tone change the answer? evaluating prompt politeness effects on modern LLMs: GPT, Gemini, and LLaMA. ArXiv:2512.12812.

[10] Cao, J., Ma, Y., Li, X., Ren, Q., Chen, X., 2026. Task-specific efficiency analysis: When SLMs outperform LLMs. ArXiv:2603.21389.

[11] Chakraborty, A., Dahal, C., Gupta, V., 2025. Federated retrieval-augmented generation: A systematic mapping study, in: Findings of the Association for Computational Linguistics: EMNLP 2025, pp. 7362–7374. ArXiv:2505.18906.

[12] Chen, W., Wang, J., Zhang, Y., Shi, Y., Jiang, T., et al., 2024. On-device large language models: A survey of model compression and system optimization. Preprint, doi:10.21203/rs.3.rs-7975734/v1.

[13] Chen, W., Wu, L., Hu, Y., Li, Z., Cheng, Z., Qian, Y., Zhu, L., Hu, Z., Liang, L., Tang, Q., Liu, Z., Yang, H., 2025a. AutoNeural: Co-designing vision–language models for NPU inference. ArXiv:2512.02924.

[14] Chen, Y., Qian, P., Wang, S., Zhang, S., Xu, H., Lin, S., Wei, X., 2026a. Does RAG know when retrieval is wrong? diagnosing context compliance under knowledge conflict. URL: https://arxiv.org/abs/2605.14473, arXiv:2605.14473.

[15] Chen, Z., Chen, W., Smiley, C., Shah, S., Borova, I., Langdon, D., Moussa, R., Beane, M., Huang, T.H., Routledge, B., Wang, W.Y., 2021. FinQA: A dataset of numerical reasoning over financial data, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 3697–3711.

[16] Chen, Z., Hu, Y., Fu, Z., Li, Z., Huang, J., Huang, Q., Wei, Y., 2026b. INTENT: Invariance and discrimination-aware noise mitigation for robust composed image retrieval, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 20463–20471. doi:10.1609/aaai.v40i25.39181. also arXiv:2604.18051.

[17] Chen, Z., Hu, Y., Li, Z., Fu, Z., Song, X., Nie, L., 2025b. OFFSET: Segmentation-based focus shift revision for composed image retrieval, in: Proceedings of the ACM International Conference on Multimedia, pp. 6113–6122. Also arXiv:2507.05631.

[18] Chen, Z., Hu, Y., Li, Z., Fu, Z., Wen, H., Guan, W., 2025c. HUD: Hierarchical uncertainty-aware disambiguation network for composed video retrieval, in: Proceedings of the ACM International Conference on Multimedia, pp. 6143–6152. Also arXiv:2512.02792.

[19] Cheng, Z., Lai, L., 2026. Energy-efficient on-device RAG on a mobile NPU: System design and benchmark on Snapdragon X Elite. URL: https://arxiv.org/abs/2606.11257, arXiv:2606.11257.

[20] Cheng, Z., Lai, L., Liu, Y., 2026a. Resolving the robustness-precision trade-off in financial RAG through hybrid document-routed retrieval. ArXiv:2603.26815.

[21] Cheng, Z., Lai, L., Liu, Y., Cheng, K., Qi, X., 2026b. Enhancing financial report question-answering: A retrieval-augmented generation system with reranking analysis. ArXiv:2603.16877.

[22] Choe, J., Kim, J., Jung, W., 2025. Hierarchical retrieval with evidence curation for open-domain financial question answering on standardized documents, in: Findings of the Association for Computational Linguistics: ACL 2025, pp. 16663–16681.

[23] Courty, B., Schmidt, V., Goyal-Kamal, Luccioni, S., Feld, B., Connell, L., Blank, D., Wilson, J., Friedler, S., Lannelongue, L., 2023. CodeCarbon: Estimate and track carbon emissions from machine learning computing. Software package, https://github.com/mlco2/codecarbon.

[24] Dai, Y., Chen, M., Zuo, Z., 2023. Neighbors in space: Satellite imagery and chinese B-share discount. China Economic Review 82, 102063. doi:10.1016/j.chieco.2023.102063.

[25] Dao, T., 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning, in: Proceedings of the 12th International Conference on Learning Representations (ICLR).

[26] Dao, T., Fu, D.Y., Ermon, S., Rudra, A., Ré, C., 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness, in: Advances in Neural Information Processing Systems.

[27] Dettmers, T., Lewis, M., Belkada, Y., Zettlemoyer, L., 2022. LLM.int8(): 8-bit matrix multiplication for transformers at scale, in: Advances in Neural Information Processing Systems.

[28] Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L., 2023. QLoRA: Efficient finetuning of quantized LLMs, in: Advances in Neural Information Processing Systems.

[29] Devadas, R.M., Sowmya, T., 2026. Towards carbon-aware AI: A systematic PRISMA review and taxonomy of green architectures, hardware life-cycle, and energy-efficient algorithms. Energy Informatics 9, 38. doi:10.1186/s42162-026-00651-8.

[30] Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), pp. 4171–4186.

[31] Dodge, J., Prewitt, T., Tachet des Combes, R., Odmark, E., Schwartz, R., Strubell, E., Luccioni, A.S., Smith, N.A., DeCario, N., Buchanan, W., 2022. Measuring the carbon intensity of AI in cloud instances, in: Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT), pp. 1877–1894. doi:10.1145/3531146.3533234.

[32] Es, S., James, J., Espinosa-Anke, L., Schockaert, S., 2024. RAGAS: Automated evaluation of retrieval augmented generation, in: Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, Association for Computational Linguistics, St. Julians, Malta. pp. 150–158. doi:10.18653/v1/2024.eacl-demo.16. also arXiv:2309.15217.

[33] Fan, T., Wang, J., Ren, X., Huang, C., 2025. MiniRAG: Towards extremely simple retrieval-augmented generation. ArXiv:2501.06713.

[34] Fang, Y., Sun, T., Shi, Y., Gu, X., 2025. AttentionRAG: Attention-guided context pruning in retrieval-augmented generation. ArXiv:2503.10720.

[35] Feng, C., Shen, M., Balashankar, A., Gerner-Beuerle, C., Rodrigues, M.R.D., 2026. Noisy but valid: Robust statistical evaluation of LLMs with imperfect judges, in: Proceedings of the 14th International Conference on Learning Representations (ICLR). ArXiv:2601.20913.

[36] Frantar, E., Alistarh, D., 2023. SparseGPT: Massive language models can be accurately pruned in one-shot, in: Proceedings of the 40th International Conference on Machine Learning (ICML).

[37] Frantar, E., Ashkboos, S., Hoefler, T., Alistarh, D., 2023. GPTQ: Accurate post-training quantization for generative pre-trained transformers, in: Proceedings of the 11th International Conference on Learning Representations (ICLR).

[38] Fu, R., Meng, C., Wang, Y., Zhao, H., Lu, J., Liu, K., Dou, J., Fong, S.J., 2026a. NeuroPareto: Calibrated acquisition for costly many-goal search in vast parameter spaces. ArXiv:2602.03901.

[39] Fu, R., Tan, J.Y., Meng, C., Yin, S., Ma, X., Wu, W., Qi, M., Yao, G., Kang, Z., Su, Z., Fong, S., 2026b. GaiaFlow: Semantic-guided diffusion tuning for carbon-frugal search. ArXiv:2602.15423.

[40] Fu, Z., Fan, C., Jiang, L., 2025. CO2-Meter: A comprehensive carbon footprint estimator for LLMs on edge devices. ArXiv:2511.08575.

[41] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., Wang, H., 2024. Retrieval-augmented generation for large language models: A survey. ArXiv:2312.10997.

[42] Gemma Team, Mesnard, T., Hardin, C., Dadashi, R., Bhupatiraju, S., Pathak, S., Sifre, L., Rivière, M., Kale, M.S., Love, J., et al., 2024. Gemma: Open models based on gemini research and technology. ArXiv:2403.08295.

[43] Gerganov, G., llama.cpp contributors, 2023. llama.cpp: LLM inference in C/C++. https://github.com/ggml-org/llama.cpp. Open-source software; experiments in this work use commit 6658925.

[44] Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C.C.T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., Salim, A., Shah, S., Behl, H.S., Wang, X., Bubeck, S., Eldan, R., Kalai, A.T., Lee, Y.T., Li, Y., 2023. Textbooks are all you need. ArXiv:2306.11644.

[45] Guo, Y., Guo, M., Su, J., Yang, Z., Zhu, M., Li, H., Qiu, M., Liu, S.S., 2024. Bias in large language models: Origin, evaluation, and mitigation ArXiv:2411.10915.

[46] Han, X., Liu, J., Zheng, M., Zhang, Z., Wu, C., 2026a. Interpretable versus learned encoders for high-cardinality fraud detection. URL: https://arxiv.org/abs/2607.00477, arXiv:2607.00477.

[47] Han, X., Wu, C., 2026. Validation-stage combinatorial fusion analysis for imbalanced credit-card fraud detection. URL: https://arxiv.org/abs/2606.10393, arXiv:2606.10393.

[48] Han, X., Xiao, Y., Wu, C., Zhang, T., 2026b. How early is early enough? design-dependent observation-window sufficiency in subscription churn prediction. URL: https://arxiv.org/abs/2607.00473, arXiv:2607.00473.

[49] Han, X., Xiao, Y., Zhang, Z., Zheng, M., 2026c. Interpretable factor decomposition for decision intelligence in large-scale financial markets: Evidence from china's A-share market. URL: https://arxiv.org/abs/2606.12843, doi:10.48550/arXiv.2606.12843, arXiv:2606.12843.

[50] He, Q., 2025. A unified metric architecture for AI infrastructure: A cross-layer taxonomy integrating performance, efficiency, and cost. doi:10.48550/arXiv.2511.21772. arXiv:2511.21772.

[51] He, Q., Shan, R., Qu, C., Tang, Y., Zou, Y., 2026. Rare-earth exposure and bottlenecks in AI data centers: Cost and schedule risk. URL: https://ssrn.com/abstract=6232318. sSRN 6232318.

[52] Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., 2022. LoRA: Low-rank adaptation of large language models, in: Proceedings of the 10th International Conference on Learning Representations (ICLR).

[53] Hu, L., Xin, Y., Shen, B., Cai, H., Jin, L., 2026a. CoDES: A context-efficient framework for enhancing small language models via domain-specific adaptation and model ensembling. doi:10.20944/preprints202603.1152.v1. preprints.org 2026.

[54] Hu, M., Wang, J., Zhao, W., Zeng, Q., Luo, L., 2025. FlowMalTrans: Unsupervised binary code translation for malware detection using flow-adapter architecture. ArXiv:2508.20212.

[55] Hu, Y., Li, Z., Chen, Z., Huang, Q., Fu, Z., Xu, M., Nie, L., 2026b. REFINE: Composed video retrieval via shared and differential semantics enhancement. ACM Transactions on Multimedia Computing, Communications, and Applications doi:10.1145/3796712.

[56] Ignatov, A., Timofte, R., Chou, W., Wang, K., Wu, M., Hartley, T., Van Gool, L., 2019. AI benchmark: Running deep neural networks on android smartphones, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops.

[57] Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.A., Stock, P., Le Scao, T., Lavril, T., Wang, T., Lacroix, T., El Sayed, W., 2023. Mistral 7B. ArXiv:2310.06825.

[58] Jiang, D., Li, Y., Li, G., Li, B., 2026a. MAGMA: A multi-graph based agentic memory architecture for AI agents. ArXiv:2601.03236.

[59] Jiang, D., Li, Y., Wei, S., Yang, J., Kishore, A., Zhao, A., Kang, D., Hu, X., Chen, F., Li, Q., Li, B., 2026b. Anatomy of agentic memory: Taxonomy and empirical analysis of evaluation and system limitations. ArXiv:2602.19320.

[60] Jiang, X., Yang, S., Yang, W., Liu, Y., Ji, C., 2026c. Agentic AI as a cybersecurity attack surface: Threats, exploits, and defenses in runtime supply chains. ArXiv:2602.19555.

[61] Jiang, Y., Li, D., Ferraro, F., 2026d. DRP: Distilled reasoning pruning with skill-aware step decomposition for efficient large reasoning models. URL: https://arxiv.org/abs/2505.13975, arXiv:2505.13975.

[62] Jiang, Y., Li, R., Dipta, S.R., Li, D., Yang, Z., 2026e. Cornerstones or stumbling blocks? deciphering the rock tokens in on-policy distillation. URL: https://arxiv.org/abs/2605.09253, arXiv:2605.09253.

[63] Jin, J., Zhu, Y., Dong, G., Zhang, Y., Yang, X., Zhang, C., Zhao, T., Yang, Z., Dou, Z., Wen, J.R., 2025. FlashRAG: A modular toolkit for efficient retrieval-augmented generation research, in: Companion Proceedings of the ACM Web Conference 2025, ACM, Sydney, NSW, Australia. pp. 737–740. doi:10.1145/3701716.3715313. also arXiv:2405.13576.

[64] Johnson, J., Douze, M., Jégou, H., 2021. Billion-scale similarity search with GPUs. IEEE Transactions on Big Data 7, 535–547. doi:10.1109/TBDATA.2019.2921572.

[65] Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., McHardy, R., 2023. Challenges and applications of large language models. ArXiv:2307.10169.

[66] Kallakuri, U., Humes, E., Jonna, R., Lin, X., Mohsenin, T., 2025. Enabling on-device medical AI assistants via input-driven saliency adaptation. ArXiv:2506.11105.

[67] Kang, D.H., Lee, H., Cha, H., Choi, M., Lim, S., 2026. Pocket RAG: On-device RAG for first aid guidance in offline mobile environments. ArXiv:2602.13229.

[68] Kaplan, J., McCandlish, S., Henighan, T., Brown, T.B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., Amodei, D., 2020. Scaling laws for neural language models. ArXiv:2001.08361.

[69] Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., Yih, W.t., 2020. Dense passage retrieval for open-domain question answering, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6769–6781.

[70] Kim, J., Um, H., Kim, S., Park, Y., Seo, J., 2025. FlexiQ: Adaptive mixed-precision quantization for latency/accuracy trade-offs in deep neural networks. ArXiv:2510.02822.

[71] Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T., 2019. Quantifying the carbon emissions of machine learning. ArXiv:1910.09700.

[72] Lai, L., Cheng, Z., Cheng, K., Qi, X., 2026. Do transformers always win? an empirical study of semantic embeddings for short-text e-commerce reviews. Research Square Preprint, doi:10.21203/rs.3.rs-9163424/v1.

[73] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D., 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Advances in Neural Information Processing Systems, pp. 9459–9474.

[74] Li, D., Jiang, B., Huang, L., Beigi, A., Zhao, C., Tan, Z., Bhattacharjee, A., Jiang, Y., Chen, C., Wu, T., Shu, K., Cheng, L., Liu, H., 2025a. From generation to judgment: Opportunities and challenges of LLM-as-a-judge, in: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pp. 2757–2791. doi:10.18653/v1/2025.emnlp-main.138. also arXiv:2411.16594.

[75] Li, X., Ma, Y., Huang, Y., Wang, X., Lin, Y., Zhang, C., 2024. Synergized data efficiency and compression (SEC) optimization for LLMs, in: Proceedings of the 2024 IEEE 6th International Conference on Electronics Information and Emergency Communication (EIECS). IEEE 10800533.

[76] Li, Y., Fan, Z., Zhuang, Z., 2026a. Auditing reasoning-trace memorization claims after unlearning with head-conditioned canaries. URL: https://arxiv.org/abs/2605.18891, arXiv:2605.18891.

[77] Li, Y., Fan, Z., Zhuang, Z., 2026b. SafetyRepro: Configuration-conditional rank instability on alignment benchmarks. URL: https://arxiv.org/abs/2605.25492, arXiv:2605.25492.

[78] Li, Z., Chen, Z., Wen, H., Fu, Z., Hu, Y., Guan, W., 2025b. ENCODER: Entity mining and modification relation binding for composed image retrieval, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 5101–5109. doi:10.1609/aaai.v39i5.32541.

[79] Li, Z., Fu, Z., Hu, Y., Chen, Z., Wen, H., Nie, L., 2025c. FineCIR: Explicit parsing of fine-grained modification semantics for composed image retrieval. ArXiv:2503.21309.

[80] Li, Z., Hu, Y., Chen, Z., Huang, Q., Qiu, G., Fu, Z., Liu, M., 2026c. ReTrack: Evidence-driven dual-stream directional anchor calibration network for composed video retrieval, in: Proceedings of the 40th AAAI Conference on Artificial Intelligence (AAAI).

[81] Li, Z., Hu, Y., Chen, Z., Zhang, S., Huang, Q., Fu, Z., Wei, Y., 2026d. HABIT: Chrono-synergia robust progressive learning framework for composed image retrieval, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 6762–6770. doi:10.1609/aaai.v40i8.37608. also arXiv:2604.18037.

[82] Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., et al., 2023. Holistic evaluation of language models. Transactions on Machine Learning Research ISSN 2835-8856.

[83] Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.M., Wang, W.C., Xiao, G., Dang, X., Gan, C., Han, S., 2024. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration, in: Proceedings of Machine Learning and Systems (MLSys).

[84] Liu, M., Shi, G., 2024. Enhancing LLM-based text classification in political science: Automatic prompt optimization and dynamic exemplar selection for few-shot learning. ArXiv:2409.01466.

[85] Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P., 2024a. Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics 12, 157–173.

[86] Liu, Q., Huang, D., Zapater, M., Atienza, D., 2025a. GreenLLM: SLO-aware dynamic frequency scaling for energy-efficient LLM serving. ArXiv:2508.16449.

[87] Liu, Y., Wu, J., He, Y., Gong, R., Xia, J., Li, L., Gao, H., Chen, H., Bi, B., Zhang, J., Huang, Z., Hooi, B., Li, S.Z., Li, K., 2025b. Efficient inference for large reasoning models: A survey. ArXiv:2503.23077.

[88] Liu, Y., Zhang, J., Dai, Y., 2023. Analyst following and greenwashing decision. Finance Research Letters 58, 104510. doi:10.1016/j.frl.2023.104510.

[89] Liu, Z., Zhao, C., Iandola, F., Lai, C., Tian, Y., Fedorov, I., Xiong, Y., Chang, E., Shi, Y., Krishnamoorthi, R., Lai, L., Chandra, V., 2024b. MobileLLM: Optimizing sub-billion parameter language models for on-device use cases, in: Proceedings of the 41st International Conference on Machine Learning (ICML).

[90] Lu, Y., Chen, X., Gu, J., Zhang, Y., Xuan, Q., Zhu, Z., 2025. Dataset distillation with pre-trained models: A contrastive approach. Neurocomputing doi:10.1016/j.neucom.2025.132015.

[91] Lu, Y., Cheng, H., Fang, Y., Wang, Z., Wei, J., Xu, D., Xuan, Q., Zhu, Z., 2026. Reassessing layer pruning in large language models, in: Proceedings of the 14th International Conference on Learning Representations (ICLR). ArXiv:2411.15558.

[92] Lu, Y., Zhu, Y., Li, Y., Xu, D., Lin, Y., Xuan, Q., Yang, X., 2024. A generic layer pruning method for signal modulation recognition deep networks. IEEE Transactions on Cognitive Communications and Networking ArXiv:2406.07929; IEEE Xplore:10810366.

[93] Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J., Wei, F., 2024. The era of 1-bit LLMs: All large language models are in 1.58 bits. ArXiv:2402.17764.

[94] Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., Hajishirzi, H., 2024. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL). ArXiv:2310.01558.

[95] Muennighoff, N., Tazi, N., Magne, L., Reimers, N., 2023. MTEB: Massive text embedding benchmark, in: Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp. 2014–2037.

[96] Muttur, R.C., 2025. SRAS: A lightweight reinforcement learning-based document selector for edge-native RAG pipelines. ArXiv:2601.01785.

[97] Nissen, L., Zagar, P., Ravi, V., Zahedivash, A., Aalami, O., Schmiedmayer, P., Reimer, L.M., Jonas, S., 2025. Medicine on the edge: Comparative performance analysis of on-device LLMs for clinical reasoning. ArXiv:2502.08954.

[98] Page, M.J., McKenzie, J.E., Bossuyt, P.M., Boutron, I., Hoffmann, T.C., Mulrow, C.D., Shamseer, L., Tetzlaff, J.M., Akl, E.A., Brennan, S.E., Chou, R., Glanville, J., Grimshaw, J.M., Hróbjartsson, A., Lalu, M.M., Li, T., Loder, E.W., Mayo-Wilson, E., McDonald, S., Moher, D., 2021. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. BMJ 372, n71. doi:10.1136/bmj.n71.

[99] Park, T., Lee, G., Kim, M.S., 2025. MobileRAG: A fast, memory-efficient, and energy-efficient method for on-device RAG. ArXiv:2507.01079.

[100] Qi, Y., Zhao, S., Xue, Z., Zeng, X., Yu, Z., 2026. Detecting abnormal user feedback patterns through temporal sentiment aggregation. ArXiv:2604.00020.

[101] Qian, P., Wang, S., Wang, X., Chen, Y., Xu, W., Yu, Q., Lin, S., Zhang, S., You, J., Wei, X., 2026. Relevant is not warranted: Evidence-force calibration for cited RAG. URL: https://arxiv.org/abs/2605.28044, arXiv:2605.28044.

[102] Qin, X., Yu, R., Khayati, A., Qiu, Z., Zou, G., Li, Y., Wang, L., 2025. Interpretable and interactive deep survival analysis with time-dependent EXtreme gradient integration, in: 2025 IEEE International Conference on Data Mining (ICDM), IEEE. pp. 673–682.

[103] Reddi, V.J., Cheng, C., Kanter, D., Mattson, P., Schmuelling, G., Wu, C.J., Anderson, B., Breughe, M., Charlebois, M., Chmielewski, W., et al., 2020. MLPerf inference benchmark, in: Proceedings of the ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA), pp. 446–459. doi:10.1109/ISCA45697.2020.00045.

[104] Rojahn, M., Grum, M., 2025. Green AI: A systematic review and meta-analysis of its definitions, lifecycle models, hardware and measurement attempts. ArXiv:2511.07090.

[105] Sanh, V., Debut, L., Chaumond, J., Wolf, T., 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, in: Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC^2), NeurIPS 2019 Workshop. Also arXiv:1910.01108.

[106] Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O., 2020. Green AI. Communications of the ACM 63, 54–63. doi:10.1145/3381831.

[107] Seemakhupt, K., Liu, S., Khan, S., 2024. EdgeRAG: Online-indexed RAG for edge devices. ArXiv:2412.21023.

[108] Sharma, C., 2025. A comprehensive survey on retrieval-augmented generation. ArXiv:2506.00054.

[109] Sharma, R., Mehta, M., 2025. Small language models for agentic systems: A survey. ArXiv:2510.03847.

[110] Shi, Y., Sun, M., Liu, Z., Yang, M., Fang, Y., Sun, T., Gu, X., 2026. Reasoning in trees: Improving retrieval-augmented generation for multi-hop question answering. ArXiv:2601.11255.

[111] Strubell, E., Ganesh, A., McCallum, A., 2019. Energy and policy considerations for deep learning in NLP, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 3645–3650.

[112] Su, J., Lan, Q., Xia, Y., Sun, L., Tian, W., Shi, T., He, L., Jingsong, Y., 2026. Difficulty-aware agentic orchestration for query-specific multi-agent workflows, in: Proceedings of the ACM Web Conference 2026 (WWW '26), pp. 1–12. doi:10.1145/3774904.3792240.

[113] Sun, X., Gao, W., Wang, Y., Kong, L., Li, Y., Fan, Z., Zhuang, Z., Dong, W., Zheng, Z., Paranjape, H., Mandal, A., Zhang, J.R., 2026. Beyond accuracy: Measuring bias acknowledgment in chain-of-thought reasoning for responsible AI evaluation. URL: https://arxiv.org/abs/2606.15127, arXiv:2606.15127. iCML Workshop on AI4GOOD.

[114] Sun, Z., Yu, H., Song, X., Liu, R., Yang, Y., Zhou, D., 2020. MobileBERT: a compact task-agnostic BERT for resource-limited devices, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 2158–2170.

[115] Tan, F., Lee, R., Dudziak, Ł., Hu, S.X., Bhattacharya, S., Hospedales, T., Tzimiropoulos, G., Martinez, B., 2024. MobileQuant: Mobile-friendly quantization for on-device language models, in: Findings of the Association for Computational Linguistics: EMNLP 2024.

[116] Tay, Y., Dehghani, M., Bahri, D., Metzler, D., 2022. Efficient transformers: A survey. ACM Computing Surveys 55, 1–28. doi:10.1145/3530811.

[117] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., Lample, G., 2023. LLaMA: Open and efficient foundation language models. ArXiv:2302.13971.

[118] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need, in: Advances in Neural Information Processing Systems.

[119] Verdecchia, R., Sallou, J., Cruz, L., 2023. A systematic review of Green AI. WIREs Data Mining and Knowledge Discovery 13, e1507. doi:10.1002/widm.1507.

[120] de Vries, A., 2023. The growing energy footprint of artificial intelligence. Joule 7, 2191–2194. doi:10.1016/j.joule.2023.09.004.

[121] Wang, F., Zhang, Z., Zhang, X., Wu, Z., Mo, T., Lu, Q., Wang, W., Li, R., Xu, J., Tang, X., He, Q., Ma, Y., Huang, M., Wang, S., 2024. A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with LLMs, and trustworthiness. ArXiv:2411.03350.

[122] Wang, H., Ma, S., Dong, L., Huang, S., Wang, H., Ma, L., Yang, F., Wang, R., Wu, Y., Wei, F., 2023. BitNet: Scaling 1-bit transformers for large language models. ArXiv:2310.11453.

[123] Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., Zhou, M., 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, in: Advances in Neural Information Processing Systems. Also arXiv:2002.10957.

[124] Wang, X., Dang, T., Zhang, X., Kostakos, V., Witbrock, M.J., Jia, H., 2025a. HealthSLM-Bench: Benchmarking small language models for mobile and wearable healthcare monitoring. ArXiv:2509.07260.

[125] Wang, X., Tang, Z., Guo, J., Liu, Z., Xia, J., Xu, M., et al., 2025b. Empowering edge intelligence: A comprehensive survey on on-device AI models. ACM Computing Surveys 57. doi:10.1145/3724420.

[126] Wang, Y., Shi, Y., Yang, M., Zhang, R., He, S., Lian, H., Chen, Y., Ye, S., Cai, K., Gu, X., 2026a. SWE-Pruner: Self-adaptive context pruning for coding agents. ArXiv:2601.16746.

[127] Wang, Y., Tang, Y., Qian, Y., Zhao, C., 2026b. VisualLeakBench: Auditing the fragility of large vision-language models against PII leakage and social engineering. ArXiv:2603.13385.

[128] Wang, Z., Wang, Z., Le, L.T., Zheng, H.S., Mishra, S., Perot, V., Zhang, Y., Mattapalli, A., Taly, A., Shang, J., Lee, C.Y., Pfister, T., 2025c. Speculative RAG: Enhancing retrieval augmented generation through drafting, in: Proceedings of the 13th International Conference on Learning Representations (ICLR).

[129] Wu, C., 2026. Class weighting versus amount conditioning in credit-card fraud detection: A dollar-metric study with a temporal explanation audit. URL: https://arxiv.org/abs/2607.14686, arXiv:2607.14686.

[130] Wu, C., Lin, Y., 2026. Evidence-unit fairness and the limits of query-adaptive sparse-dense fusion in financial document retrieval. URL: https://arxiv.org/abs/2608.00183, arXiv:2608.00183.

[131] Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., Han, S., 2023. SmoothQuant: Accurate and efficient post-training quantization for large language models, in: Proceedings of the 40th International Conference on Machine Learning (ICML).

[132] Xu, J., Li, Z., Chen, W., Wang, Q., Gao, X., Cai, Q., Ling, Z., 2024. On-device language models: A comprehensive review. ArXiv:2409.00088.

[133] Yao, C., Zhan, Q., Cao, Z., Li, D., Lin, Y., Shao, Y., Wang, L., Wang, Z., Zhang, J., Zhang, Y., Zhao, S., Yao, Z., Xu, H., Dong, J., Wang, S., Yu, C.H., Dong, Z., 2025. Generative AI for simulating real world dynamics: Applications and challenges. doi:10.36227/techrxiv.176049719.90048379. techRxiv preprint.

[134] Yin, Z., Zhang, M., Kawahara, D., 2024. Harmony: A smart home assistant with local LLM. ArXiv:2410.14252.

[135] You, J., Chung, J.W., Chowdhury, M., 2023. Zeus: Understanding and optimizing GPU energy consumption of DNN training, in: Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI).

[136] Zeng, Y., Yu, W., Li, Z., Ren, T., Ma, Y., Cao, J., Chen, X., Yu, T., 2025. Bridging the editing gap in LLMs: FineEdit for precise and targeted text modifications, in: Findings of the Association for Computational Linguistics: EMNLP 2025. doi:10.18653/v1/2025.findings-emnlp.118. also arXiv:2502.13358.

[137] Zhan, Q., Sun, D., Gao, E., Ma, Y., Liang, Y., Yang, H., 2024. Advancements in feature extraction recognition of medical imaging systems through deep learning technique. ArXiv:2406.18549.

[138] Zhang, H., Yang, S., Liang, X., Shang, C., Jiang, Y., Tao, C., Xiong, J., So, H.K.H., Xie, R., Chang, A.X., Wong, N., 2026a. Find your optimal teacher: Personalized data synthesis via router-guided multi-teacher distillation. URL: https://arxiv.org/abs/2510.10925, arXiv:2510.10925.

[139] Zhang, J.R., Du, G., Sun, Q., Wang, S., Li, J., Sun, X., 2026b. Carbon-aware compute–power scheduling for AI data centers with microgrid prosumer operations, in: IEEE International Conference on Research and Technologies for Society and Industry, IEEE. Also arXiv:2605.03751.

[140] Zhang, P., Zeng, G., Wang, T., Lu, W., 2024. TinyLlama: An open-source small language model. ArXiv:2401.02385.

[141] Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.Y., Wen, J.R., 2023. A survey of large language models. ArXiv:2303.18223.

[142] Zheng, D., Fu, W., 2025. Carbon- and system-aware LoRA scaling for on-device LLMs via hierarchical multi-objective reinforcement learning. Preprint, presented at NeurIPS 2025 Workshop.

[143] Zheng, Y., Chen, Y., Qian, B., Shi, X., Shu, Y., Chen, J., 2025. A review on edge large language models: Design, execution, and applications. ACM Computing Surveys doi:10.1145/3719664.

[144] Zhu, F., Lei, W., Huang, Y., Wang, C., Zhang, S., Lv, J., Feng, F., Chua, T.S., 2021. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 3277–3287.

[145] Zhu, X., Li, J., Liu, Y., Ma, C., Wang, W., 2024. A survey on model compression for large language models. Transactions of the Association for Computational Linguistics 12, 1556–1577.

[146] Zhu, Y., Gu, J.C., Sikora, C., Ko, H., Liu, Y., Lin, C.C., Shu, L., Luo, L., Meng, L., Liu, B., Chen, J., 2025. Accelerating inference of retrieval-augmented generation via sparse context selection, in: Proceedings of the 13th International Conference on Learning Representations (ICLR).

[147] Zhuang, Z., Li, Y., Fan, Z., 2026. Pre-registering the detectable effect: A paired-MDE budget for 4-bit quantization benchmarks, with a pilot audit. URL: https://arxiv.org/abs/2605.28873, arXiv:2605.28873.