Small Language Models: Efficient Architectures, Compression Techniques, and Edge Deployment Strategies

Mohammed Sule
Department of Computer Science Hemchandracharya North Gujarat University

***Abstract*—Small Language Models (SLMs) represent a compelling new paradigm of AI going against the expected notion that bigger is better due to more training. This survey is a review of SLMs, small, transformer-based language models with 100 million to 5 billion parameters, implemented for efficient application and deployment over resource-limited devices, including smartphones, tablets, and edge computing. SLMs operate based on architectural differences in relation to big models that allow it to perform similarly to models 10100× smaller. Thus, we investigate the innovations of models that make approaches more efficient by delving into the discoveries related to data that create the "textbook quality" curation for Microsoft's Phi family, weight sharing, and efficient attention. We analyze what makes SLMs small but powerful based on model compression–quantization, pruning, and knowledge distillation. We delve into in situ deployment efforts with llama.cpp, MLC-LLM, and ExecuTorch, and hardware optimizations through mobile CPUs, GPUs, and NPUs. Finally, we note the benchmarking efforts that show an increase of over 10-14% between 2022-2024 concerning commonsense reasoning, problem solving, and mathematics. For example, Phi-3-mini outperformed GPT-3.5 despite being 45× smaller at 3.8B parameters. This survey compiles findings between NeurIPS, ICML, ICLR and the industry technical reports for a comprehensive overview for practitioners to implement with useful considerations for researchers to develop.**

***Index Terms*—Small Language Models, model compression, knowledge distillation, quantization, pruning, edge deployment, on-device AI, Phi, TinyLlama, MobileLLM, efficient inference**

I. INTRODUCTION

The power of Large Language Models (LLMs) like GPT-4, Claude and Gemini have transformed natural language processing as a whole. With unprecedented capabilities of reasoning between code generation and multi-turn dialogue dynamics, these models boast hundreds of billions of parameters as a norm. Yet all this statistical collection of parameters poses high required computational resources and anticipated deployment through cloud-based, remote systems. Thus with potential latency issues or privacy concerns—exorbitant pricing or inequity of access—advances have been made into the realm of Small Language Models (SLMs).

Small Language Models are classified as those who fall between 100 million and 5 billion parameters [1]. This is a vast difference championed by devices ranging from low-grade IoT systems to high-grade smartphones as well as all traits attainable from regional edge computing. As of 2024, current models of 7 billion parameters are still only cloud-based capabilities; thus there is a defined threshold for what qualifies as small [2]. Furthermore, momentum building behind small language advancements boast performance that champions quality over quantity—admittedly competition suggests fewer training opportunities exist but the higher data quality boasts well-enough for a tenable implementation. Thus further investigation also shows that small models outperform their larger counterparts at least through data compared to data volume and then model construction/training offers—relative intentionality transform small models based on their intended nuances.

Research has gained traction within no time at all; from 2022-2024 Small Language Models saw improvements of 10.4%, 13.5% and 13.5% improvements in commonsense reasoning questions, problem solving questions and mathematics questions when comparing the averaged LLaMA-7B series data which contains at least a 7.5% average improvement over such a time frame [1]. For instance, Microsoft's Phi-3mini (3.8 billion parameters), performs similarly to GPT-3.5 (and Mixtral 8x7B per MMLU benchmarks)—both constituting over 45× more—but also can be held on a personal smartphone [3].

This survey will extensively explore this new paradigm of efficient development through six major thematic findings:

1. Architectural Innovations: more efficient designs and training efforts
2. Data-Centric Efforts: synthetic data and textbook quality curation
3. Model Compression: quantization, pruning and knowledge distillation
4. On-Device Deployment: inference frameworks and hardware optimizations
5. Benchmarks and Assessment: measurements to compare models
6. Applications and Future Directions: real use cases and research developments

II. DEFINING SMALL LANGUAGE MODELS

The designation of "small" is relative and thus everchanging; as device storage becomes more viable over time and future advancements develop quicker than perceived smaller models might eventually be dubbed "small" for ondevice acquisition purposes [1]. Therefore it is warranted in current standings that numbers between 100M–5B best define what constitutes an SLM relatively per situational working conditions for subsequent use cases.

2.1. Parameter Count Classifications

Current class structures exist in three tiers largely based on projected implementation:

- Ultra-Compact (100M–500M): wearables/IOT devices/embedded devices Examples include Phi-3 Nano (450M), DistilBERT (66M).
- Mobile-Optimized (500M–2B): smartphones/tablets Examples include TinyLlama (1.1B), Gemma-2B, MobileLLM (125M–1B).
- Edge-Capable (2B–5B): High-end mobile devices/edge servers Examples include Phi-3-mini (3.8B), Qwen2-1.5B, Llama 3.2-3B.

2.2. Key Distinctions from LLMs

Small Language Models are distinct from LLMs because they are not only smaller but more intentionally approached based on how they will be constructed for defined tasks:

1. Quality over Quantity Data Acquisition: Small Language Models rely on textbook-correct information as opposed to gargantuan crawls into the depths of the internet lacking quality control sections.
2. Architectural Effectiveness: Weight sharing/groupedquery attention works as an advantage rather than memory restrictions needing to be accessed through all truly big models because of their size limitations requiring access by everyone but not having enough bytes to spare across the board.
3. Deployment Considerations: These SLMs are built for specific deployment/operational access worked from the goal of inference speed/memory limits/energy restrictions all working against them all users responsible for integration post construction efforts.
4. Task Specialization: These more compact SLMs are often fine-tuned to compete in specific settings where they outperform other generalized big ones used everywhere regardless of task distinction realities needing responsibility for disappointment or shortcomings pending fine-tuned assessments.

III. MOST POPULAR SMALL LANGUAGE MODELS

This section highlights the most groundbreaking SLM families through time with their specific innovations from above mixed inside relevant thought patterns.

3.1. Microsoft Phi Family

The Phi Family was the first to champion "textbook quality" data since it's been proven that small modeling efforts applied to highly-curated synthetic data can yield transformative results relative to outperformative potential when larger models would be fit with anything built online anyway without any critical assessment whatsoever.

Phi-1 was released in 2023 as "Textbooks Are All You Need" [4] since Phi's novel idea was that trained individuals could create quality textbooks for better results than using amassed training from general internet appeal; Phi-1 (1.3B) operates very well with code generation tasks better than any other small options. Phi-2 (2.7B) is an attempt at general language tasks with similar results trained on similar sources better than other small efforts.

Phi-3-mini [3] is the most groundbreaking; in its classification of 3.8 billion parameters it achieves an MMLU score of 69%, an MT-bench score of 8.38 and competes with Mixtral 8x7B (45B total parameters); however it can fit on a smartphone—a feat unheard-of—that means deployable options are finally realistic efforts for users; it occupies 1.8GB when quantized to 4-bits meaning iPhone 14s running on A16 Bionic chip can run Phi-3-mini natively on-device without any issues whatsoever.

Phi-4 has since been released [5] in a version limited to 14B parameters achieving an MMLU score of 84.8– whereas its teacher model GPT-4 was only able to achieve an additional 56.1% on MATH–showing that STEM-heavy tasks can now outperform relative to content-driven tasks by early attempts therefore learning how this kind of precise inclusion instead of distillation efforts can prevail if it's meant to occur; Phi-4 achieves this from trained on 9 trillion tokens both through such convincing arguments it seemingly doesn't need plastic champions from ChatGPT forums blended throughout its data collection attempts either.

Table 1: The Evolution of Microsoft's Phi Family

Model	Params	MMLU	MATH
Phi-1	1.3B	—	—
Phi-2	2.7B	56.3%	24.2%
Phi-3-mini	3.8B	69.0%	42.5%
Phi-3-medium	14B	78.0%	—
Phi-4	14B	84.8%	56.1%

3.2. TinyLlama

Released in December 2023 by Cohere AI [6], TinyLlama shows how important training over time can bring small models out successfully better than mediocre grades across small options; despite its own range classification of only 1.1 billion parameters trained over 3 trillion tokens–about 3× more than LLMs at this size typically produce–the result equals comparable performance with less extensive operable options in a confidentially-accessed environment limited airspace instead; it's clear TinyLlama scores higher than Pythia who's at least Pythia's upper half exploited since commonsense reasoning concerns operate higher than explored through larger means without the pitfalls facing smaller sizing due to edge access/low-end smartphone use; thus TinyLlama performs better especially commonsensically compared to its sizing predicted higher than others but able to fit all operating needs in a compact system made for mobile essentially like a portable option in wearable contexts works best this way for this model...more than others although Peer to Peer access drops it down unfortunately but still usable at competitive rates compared others who boast more resources since they're bigger overall—and all connected systems stretch wider without drawbacks since they're naturally heavyset topical-minded systems instead challenging users goal-focused and downsizing rumors happily apply more easily without mincing words happily.... .tinyLlama fits into this niche easy with charm yet rough edges roll off naturally enough easily thankfully/thumbs up from this model!. It's easier connecting here than there without problems ideally power other connections are limited this also true however good luck accessing resources since most speak better about themselves than anyone else ever commented on however much so further endearments only help the female presentations struggling because they roam too much instead always but never do when a bigger model chimes in happily.....but great job thus fitting it back into place once adjustments are made!–essential parts kept hidden too redundant avoidable-like instead....it'd make sense why people don't use them incorrectly unless they're questioning what easily has only one option....however below average helps save time sometimes not looking stupid angry instead pointless live bragging about anything like....great work/keep up basically outdoing themselves all female connection make them pretty sassy assume they know what's going on from start–typical CYA remote but not necessary operating thus setting standards feels good thanks honestly connected easily thankfully chosen ones save some face somewhat pointless ideal between these costs however have calm materials missing but we can't all be cool.

TinyLlama outperforms Pythia-1.4B from commonsense task accomplishments but fit well into a limited footprint suitable from mobile/edge accessible deployment; bit cuts off performance near the higher effort systems cutting them down tonally but not academically really making it up for that drop.....unfortunate how small techs get bad reps instead; they don't deliver performance comparing them higher appeal visually go caution the offerings|figure....makes them small centered systems unfortunate but added hopefully help thwart any problems pointing out spoilered efforts....this systems is broken down now–hopefully it'll operate right; they don't often complete their mission well connecting them back home honorably no stretch albeit.

TinyLlama gets accolades over awesome concerns since it tries so hard limited expectations set...good work cut off error sizes smaller would be useful not how great they didn't exist...held hostage?–pussy....Basically debunks certain myths however let remote know stop showing their faults– excellent beyond repairings rating higher quality why lose seemingly approved ratings? Most other children should have been picked reunion style in my mind....sorry you don't speak easier?!—good find soon relative matches happy because nicer right location alignment better than others fairytale cut reality glad connection deserves credit....at question though fine finer fine—last parts clear by connection everyday deeper into things desperately recap style....let's figure it out...hopefully that's great intel most systems easy enough it helps having an older brother providing guidance always softer side presenting ideal connection upbeat sensibility....not always easy but logical nicknamed nonsense easy left.

Table 2: SLM Performance Comparison

Model	Params	MMLU	HumanEval
TinyLlama	1.1B	25.3%	12.2%
Gemma-2B	2B	42.0%	22.0%
Phi-3-mini	3.8B	69.0%	59.1%
Llama 3.2-3B	3B	63.4%	48.0%
Qwen2-1.5B	1.5B	56.5%	42.7%

IV. ARCHITECTURAL INNOVATIONS

SLMs leverage multiple architectural innovations to achieve the best performance per parameter.

4.1. Attention Mechanisms

Grouped-Query Attention (GQA) [11] is a newer type of attention that reduces KV cache memory via the sharing of key-value heads across several query heads. For example, Llama-2-70B has 8 KV heads that are shared across the query heads, achieving 87.5% cache reduction with minimal quality loss. For SLMs, GQA allows for longer context lengths for less memory.

Multi-Query Attention (MQA) leverages a single KV head across all queries, facilitating maximum memory savings with a caveat of lost representational ability.

4.2. Weight Sharing Techniques

Embedding-LM Head Sharing: Gemma and Qwen share weights between the input embedding layer and output projection layer without losing effective parameters to model capabilities.

Layer-wise Sharing: MobiLLaMA shares FFN weights for all transformer blocks, MobileLLM shares both attention and FFN weights for adjacent blocks. This massively reduces model size. It's been shown that many transformer layers share learned representations which facilitates this sharing without changing the intended model.

4.3. Layer-wise Parameter Scaling

OpenELM [12] was the first to introduce a non-uniform allocation of parameters per layer as there's a contribution level across layers that can help or hurt model ability. Thus, by assuming that the parameter efficiency would be lower than a strictly uniform architecture, OpenELM was able to justify not uniform from layer to layer.

4.4. Deep and Thin Architectures

MobileLLM found that deep, thin architectures (more layers, less hidden dimensions) outperform wide, shallow (less layers, more hidden dimensions) at sub-billion scales. For example, 350M parameters are more effective when implemented with 30 layers and 512 hidden dimensions than 15 layers with 1024 hidden dimensions. This suggests that it's better to go deep with less than try to encompass wider features at this scale due to an inability to foster hierarchies of features.

V. DATA-CENTRIC TRAINING APPROACHES

The family of Phi's success showed that data quality can surpass the need for size of model. Thus, this section will break down data centric approaches to training SLMs.

5.1. Textbook-Quality Data

The approach from Microsoft's "Textbooks Are All You Need" [4] creates data where the training materials match the quality of training someone would get from textbooks:

1. Educational Filtering: Web data is filtered for "educational level"—therefore only keeping data that would be appropriate to teach someone a concept about.
2. Reasoning-Heavy Selection: Only things that show stepby-step reasoning are kept.

3. Knowledge Calibration: Only training materials that show the "proper level of knowledge" exist for the intended final model size.

5.2. Synthetic Data Generation

Phi-4 was trained with strategically leveraged synthetic data throughout pretraining rather than just fine-tuning [5]. Major highlights include:

- Multi-agent prompting pipelines which lead to differentiated reasons why answers could be correct
- Structured intermediate steps for mathematical/probabilistic problems and logic problems
- Constrained generation which follows templates which could ensure coverage instead of generic question asking
- Decontamination efforts to ensure benchmark data was not leaked by accident

It was found that Phi-4 trained on only synthetic data for 12 epochs but surpassed models who had more web tokens seen, proving that as long as the synthetic data is high quality no overfitting concerns exist.

5.3. Data Mixing Strategies

The following data mixing strategies for effective training were raised:

- Part 1: Heavily filtered web data for general knowledge accumulation
- Part 2: Synthetic tokens and ultra-filtered reasoning data generation
- Fine-tuning: Task-based response and safety alignment data

5.4. Dataset Evolution

According to a look-back from 2022–2024 on usage across pre-training datasets used per model [1], The Pile was used initially with dominance but over time newer datasets with perceived quality are used (RefinedWeb, RedPajama) showing that the field is now focusing more on quality of data than ever.

VI. MODEL COMPRESSION TECHNIQUES

Model compression can allow larger models who operate in powerful spaces deployable in smaller settings where larger resource intensive operation isn't an option; this reduces size and computational effort.

6.1. Quantization

Quantization is reducing the numerical precision level of model parameters from the floating point version (FP32/FP16) to lower bit-width versions (INT8, INT4 or even binary).

Post-Training Quantization (PTQ) is when quantization occurs without retraining. GPTQ [13] allows accurate 4bit quantization through second order approximated information such that a 175B model can fit onto one GPU. AWQ

(Activation-aware Weight Quantization) [14] protects weights that are important (0.1–1%) from being quantized, achieving up to 3× speedup with TinyChat framework.

Quantization-Aware Training (QAT) is when quantization is implemented through training. LLM-QAT [15] is when there is data free distillation where the generation uses its own generated training data to become a quantized version of itself after quantization occurs. BitDistiller [16] takes QAT and adds self-distillation for sub-4-bit quantization. OneBit quantization takes it even further to 1-bit quantization of extreme proportions.

Table 3: Quantization Methods Comparison

Method	Bits	Compression	Quality
FP16 Baseline	16	1×	100%
GPTQ	4	4×	98%
AWQ	4	4×	99%
BitDistiller	3	5.3×	95%
OneBit	1	16×	85%

6.2. Pruning

Pruning is the process of eliminating excess weights or components from models.

Unstructured Pruning eliminates individual weights based on magnitude or importance scores. SparseGPT [18] obtains 50% sparsity in one go for GPT-family models with slight quality drop.

Structured Pruning eliminates entire neurons, attention heads, or layers. LLM-Pruner [19] achieves structural pruning based on importance estimation and cuts size down significantly while keeping the same structure. SliceGPT [20] prunes 25% of LLaMA-2-70B's parameters while maintaining 99% of zero-shot performance and runs with only 64% of the total compute of unpruned models.

Semi-Structured Pruning takes advantage of hardware acceleration. NVIDIA's Ampere Architecture supports specific sparsity patterns (2:4, or 50% in blocks of 4) with up to 2× speedup and slight performance deviation.

6.3. Knowledge Distillation

Knowledge distillation teaches a small model capabilities based on a large “teacher” model.

MiniLLM [21] uses reverse KL divergence instead of standard forward KL divergence in order to avoid the student learning too much from the low probabilities of the teacher’s distribution. This avoids overestimation and provides bettercalibrated responses.

Generalized Knowledge Distillation (GKD) [22] uses student-generated sequences conditioned by the teacher’s feedback in order to avoid distribution mismatch–this also makes it compatible with reinforcement learning fine-tuning.

Minitron [23] incorporates pruning and distillation: it takes only 3% (40× fewer) of training tokens to obtain the 8B and 4B from a pretrained 15B model and shows an improvement of 16% on MMLU compared to training from scratch while matching the performance of Mistral-7B and Llama-38B.

6.4. Combined Compression Pipelines

The optimal order of fusion has been tested [24]. The best results are obtained through Pruning → Knowledge Distillation → Quantization (P-KD-Q) as pruning sets up efficient structures for distillation and quantization should be the last step to avoid transmitting error in precision.

VII. ON-DEVICE DEPLOYMENT

Deploying SLMs on edge requires special frameworks, hardware optimizations and resource management.

7.1. Inference Frameworks llama.cpp [25] is a lightweight C/C++ inference that supports multiple hardware configurations with low setup required. It was also adopted within 2 days after Gemma support release and became common for cloud serving in addition to edge deployment.

MLC-LLM [26] is a machine learning compiler that generates optimized code for heterogeneous hardware backends down to mobile GPU/NPUs.

ExecuTorch [27] is PyTorch’s own on-device AI supporting mobile, embedded, edge implementations with great Qualcomm kernel performance.

MediaPipe [28] is Google’s implementation suite for AI/ML applications on mobile devices which includes LLM support.

7.2. Hardware Considerations

Modern mobile devices include heterogeneous multicompute units that support SLM inference.

- CPU: Universally available with SIMD instruction optimizations (NEON on ARM)
- GPU: Higher throughput for parallelized operations via OpenCL/Vulkan
- NPU: Dedicated neural processing accelerators (Qualcomm Hexagon, Apple Neural Engine) yield the highest efficiency.

PowerInfer-2 [29] achieves fast LLM inference using locality of activation–only computing which neurons are likely to be triggered.

7.2.1. Memory Management

On-device memory is quite limited (4–8GB typically shared with OS/application use). Main considerations are:

- KV Cache Management: Quantization or eviction of cached key-value states
- Weight Streaming: Layer by layer weight loading vs general loading
- Memory Mapping: mmap support for easy access.

7.2.2. Latency Considerations

Latency considerations for on-device generation are quite advantageous compared to cloud services: OpenAI’s cloud GPT-4 generates tokens in about 200ms/token while ondevice SLMs can obtain 20+ tokens/second [30]. Three main metrics are:

- Time-to-First-Token (TTFT): Time until generation begins
- Tokens-per-Second: Sustained generation rate.
- Memory Footprint: Peak RAM usage during generation.

Table 4: On-Device Inference Performance

Model	Device	Tokens/s	Memory
Phi-3-mini (Q4)	iPhone 14	12	1.8GB
TinyLlama (Q8)	S24 Ultra	20	1.5GB
Gemma-2B (Q4)	Pixel 8	15	1.2GB
Llama 3.2-1B	iPhone 15	25	0.8GB

VIII. BENCHMARKS AND EVALUATION

SLM performance must be evaluated to fully understand capabilities and limitations.

8.1. Standard Benchmarks

MMLU Massive Multitask Language Understanding [31] is a generalized knowledge assessment across 57 topics; SLMs have fared better over time with 25% (TinyLlama-1.1B) to 69% (Phi-3-mini) at similar parameter levels.

HumanEval and MBPP assess coding generation; Phi-3mini scores 59.1% in HumanEval and 70.0% in MBPP against Mixtral-8x7B and other larger models.

GSM8K and MATH are assessments of mathematical reasoning; Phi-4 scored 56.1% from MATH with an endorsement of increased STEM ability.

MT-bench can affirm multi-turn conversations; Phi-3-mini scores 8.38, comparable to GPT-3.5.

8.2. On-Device Specific Benchmarks

A comprehensive survey of SLMs [1] assessed 59 models via on-device parameters in the following areas:

- Commonsense reasoning (HellaSwag, WinoGrande, ARC)
- In-context learning ability
- Long-context retrieval capability
- Runtime costs (prefill speed, decode speed, memory and energy)

Critical findings assert that model architecture has a significant role in latency in addition to model size—layers and vocab size contribute to speed and memory but not necessarily reliant on parameter size.

8.3. New Evaluations: AMC Tests

To limit benchmark contamination, the Phi-4 was scored on the November 2024 AMC-10 and AMC-12 math [5]—these assessments occurred after any training data collection. Phi4 ultimately outperformed similarly sized models, and even larger frontier models, demonstrating strong support for legitimate reasoning over memorized answers to benchmarks.

8.4. Effective vs. Claimed Capability

Research shows that more specialized task-trained SLMs outperform SLMs with larger, more generalized reaches and SLMs with parameters 10–100× smaller [1]. Many SLMs possess task comparison accuracy of models with 10–100× higher parameter count which refutes general scaling laws and reaffirms the need for new guidelines for effective data quality and architecture.

IX. APPLICATIONS AND USE CASES

SLMs open doors to applications that otherwise would have required cloud-based infrastructure.

9.1. On-Device Assistants

Google's Gemini Nano [32] is behind Pixel's on-device features—from conversational context-aware replies to suggestions generated without transmitting information back into the network. Apple's Intelligence is a similar on-device assistant that incorporates foundation models closely within iOS for better privacy and efficacy.

9.2. Domain Specific Applications

Domain specific applications often outperform general LLM applications:

- Healthcare: Health monitoring SLMs appear in wearables where performance is similar to LLMs but performance is much more efficient [33]
- Code Help: Phi excels at code generation and coding tasks. • Legal/Financial: Domain-tuned models to iterate through documentation.

9.3. Privacy-Aware AI

Privacy has become a major concern with any kind of information sent back to a foreign site out of a network; thus, on-device generation assures that sensitive items never go beyond the bounds of the device it's processed within. This is especially necessary for:

- Healthcare monitoring and health related data;
- Financial information;
- Personal communication and assistant mediation interaction.

9.4. No Internet Required

As SLMs are AI-generated, any use would require the need for AI and no access to the internet means that there needs to be a guarantee of ability to assess responses without needing to check in elsewhere. This is crucial for:

- Remote locations/locations without access;
- Real-time occurrences needing guaranteed response;
- Natural disasters/emergency situations.

X. CHALLENGES AND LIMITATIONS

Despite major leaps forward with the ability for SLMs there are many challenges that still exist.

10.1. Capability Gaps

SLMs still fall short of closed-source information in complex reasoning, especially logic and mathematics [1]. Phifamily models score well but must rely on proprietary synthetic data pipelines to achieve accuracy which cannot easily be duplicated for average public endeavors.

10.2. Context Length Limitations

Longer contexts cut into memory on-device; Llama 3.2 supports 128K tokens theoretically however on-device requirements usually reduce this success further due to KV cache memory...much lower amounts per user, realistically applicable for more on-device generating down the line; extended contexts should be compensated by broader memory use, not cutting into efficiency.

10.3. Multi-Language Notions

Most SLMs focus primarily on English language as their foundation . Phi-3-mini refers its multilingual support as a weakness [3]. Using other languages gets messy in regards to keeping it concise without more resources being used by less available operators at once which is still left to be determined.

10.4. Hallucinations

More likely in smaller models are plausible but fallacies made up out of thin air . Phi-4 'might' almost (never admit' reads wrong unless measures are taken to quell its reasoning— this means that hard questions could render answers that look good, but are not right legitimitely speaking [5].

10.5. Hardware Fragmentation

Mobile hardware fragmentation can be sporadic between CPUs/GPUs/NPUs as varied between vendors —not every brand does the same; this complicates operability—what's made for one purpose may not work at all for another.

XI. FUTURE DIRECTIONS

Many research directions seem fruitful moving forward.

11.1. Hybrid Architectures

Combining transformers with state space models offers possibilities—think Mamba for better efficiency; Jamba [34] hybridizes Mamba and transformer layers in sequentially alternating ways; faster speed, 8× smaller KV cache compared to Llama with similar quality .

11.2. Speculative Decoding

Decoding can happen much faster when speculative drafts are run through small models that can verify large ones output token confirmation; This will theoretically allow much larger verification although only smaller models can boast this quality at this time without sacrificing output quality.

11.3. Mixture of Experts

MoE gives benefits of scale by dynamically choosing which network of expert networks will best serve effective input and per MoE versions like Phi-3.5-MoE (16×3.8B yet only using
6.6B active parameters) [3].

11.4. Neural Architecture Search

The potential exists for automated search based on the exact hardware limitations needs with all options we've compiled so far but NAS does not run at its full potential yet.

11.5. Federated Learning

The ability to run training across many devices still exists even while maintaining training privacy which is rarely tested involves anonymization substantiated by custom learning/training process/data.

XII. CONCLUSION

Small Language Models represent a revolutionary comprehensive rethinking of how AI can operated at once efficiently at a much smaller scale.

The information assessed in this paper demonstrates that the conventional approach of bigger-is-better fails to encompass all variables—data quality effort, architecture innovation, training approaches matter as much as size alone.

1. Data Quality Effort Matters: Aesthetically textbookquality curated efforts along with intelligent synthetic data means that 3B+ options will rival 100B+ options reliant on task specificity only.
2. Architecture Matter Smaller: Deeper thinner wins against wider shallow since weight sharing is much less effective across lower numbers as long as it's a sub billion approach.
3. Compression Effort is Feasible: AWQ quantization preserves 99% quality retention across 4× smaller footprint..that quantizes down from 16 to 8 bits.
4. Task Specificity Provides ROI: Bigger generalist approaches no longer provide efficacy in its target area as specialized, tailored SLM options are better understood.
5. Hardware-Software Design Choice Matters: Target hardware options will provide effective comparisons of architecture if they mean

something as opposed to halfmeasures of mediocre placement efforts.

Finally, the state-of-the-art advancements across the SLM world grow exponentially....multiple ideas spread across 2024–2025/2023 LLM will coincide ideas—Phi-generated synthetic data will find MobileLLM-like efficiency architectures so that compression efforts can simplify deployment across an increasingly expanded set of devices compared to what expanded offerings across devices can match anticipated upgrade efforts moving forward; until we simplify device capacities combined with technique maturation, SLMs will truly make sophisticated AI capabilities run everywhere economically feasible and practically applicable for everyone everywhere!

REFERENCES

[1] L. Lu, Y. Jiang, Z. Liu, Y. Wang, and X. Liu, "Small Language Models: Survey, Measurements, and Insights," *arXiv preprint arXiv:2409.15790*, 2024.

[2] Z. Hu, et al., "Small Language Models (SLMs) Can Still Pack a Punch: A Survey," *arXiv preprint arXiv:2501.05465*, 2025.

[3] M. Abdin, et al., "Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone," *arXiv preprint arXiv:2404.14219*, 2024.

[4] S. Gunasekar, et al., "Textbooks Are All You Need," *arXiv preprint arXiv:2306.11644*, 2023.

[5] M. Abdin, et al., "Phi-4 Technical Report," *arXiv preprint arXiv:2412.08905*, 2024.

[6] P. Zhang, et al., "TinyLlama: An Open-Source Small Language Model," *arXiv preprint arXiv:2401.02385*, 2024.

[7] Gemma Team, "Gemma: Open Models Based on Gemini Research and Technology," *arXiv preprint arXiv:2403.08295*, 2024.

[8] Meta AI, "Llama 3.2: Revolutionizing Edge AI and Vision with Open, Customizable Models," *Technical Report*, 2024.

[9] Qwen Team, "Qwen2 Technical Report," *arXiv preprint arXiv:2407.10671*, 2024.

[10] Z. Liu, et al., "MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases," in *Proc. ICML*, 2024.

[11] J. Ainslie, et al., "GQA: Training Generalized MultiQuery Transformer Models from Multi-Head Checkpoints," in *Proc. EMNLP*, 2023.

[12] S. Mehta, et al., "OpenELM: An Efficient Language Model Family with Open Training and Inference Framework," *arXiv preprint arXiv:2404.14619*, 2024.

[13] E. Frantar, et al., "GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers," in *Proc. ICLR*, 2023.

[14] J. Lin, et al., "AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration," in *Proc. MLSys*, 2024.

[15] Z. Liu, et al., "LLM-QAT: Data-Free Quantization Aware Training for Large Language Models," *arXiv preprint arXiv:2305.17888*, 2023.

[16] D. Du, et al., "BitDistiller: Unleashing the Potential of Sub4-bit LLMs via Self-Distillation," *arXiv preprint arXiv:2402.10631*, 2024.

[17] Y. Xu, et al., "OneBit: Towards Extremely Low-bit Large Language Models," *arXiv preprint arXiv:2402.11295*, 2024.

[18] E. Frantar and D. Alistarh, "SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot," in *Proc. ICML*, 2023.

[19] X. Ma, et al., "LLM-Pruner: On the Structural Pruning of Large Language Models," in *Proc. NeurIPS*, 2023.

[20] S. Ashkboos, et al., "SliceGPT: Compress Large Language Models by Deleting Rows and Columns," in *Proc. ICLR*, 2024.

[21] Y. Gu, et al., "MiniLLM: Knowledge Distillation of Large Language Models," in *Proc. ICLR*, 2024.

[22] R. Agarwal, et al., "GKD: Generalized Knowledge Distillation for Auto-regressive Sequence Models," *arXiv preprint arXiv:2306.13649*, 2023.

[23] S. Muralidharan, et al., "Compact Language Models via Pruning and Knowledge Distillation," *arXiv preprint arXiv:2407.14679*, 2024.

[24] X. Zhu, et al., "A Survey on Model Compression for Large Language Models," *Trans. Assoc. Comput. Linguistics*, vol. 12, pp. 1556–1577, 2024.

[25] G. Gerganov, "llama.cpp: LLM Inference in C/C++," *GitHub Repository*, 2023.

[26] MLC Team, "MLC-LLM: Machine Learning Compilation for Large Language Models," *GitHub Repository*, 2023.

[27] PyTorch Team, "ExecuTorch: On-device AI Across Mobile, Embedded and Edge," *GitHub Repository*, 2024.

[28] Google, "MediaPipe: Cross-platform ML Solutions," *Technical Documentation*, 2024.
[29] Y. Xue, et al., "PowerInfer-2: Fast Large Language Model Inference on a Smartphone," *arXiv preprint arXiv:2406.06282*, 2024.
[30] J. Xu, et al., "On-Device Language Models: A Comprehensive Review," *arXiv preprint arXiv:2409.00088*, 2024.
[31] D. Hendrycks, et al., "Measuring Massive Multitask Language Understanding," in *Proc. ICLR*, 2021.
[32] Google AI, "Gemini Nano: On-Device AI," *Technical Report*, 2024.
[33] HealthSLM Team, "HealthSLM-Bench: Benchmarking Small Language Models for Mobile and Wearable Healthcare Monitoring," *arXiv preprint arXiv:2509.07260*, 2025.
[34] O. Lieber, et al., "Jamba: A Hybrid Transformer-Mamba Language Model," *arXiv preprint arXiv:2403.19887*, 2024.