

An Evaluation of LLMs Inference on Popular Single-board Computers

Tung (Thomas) Nguyen¹*, Tuyen Nguyen²

¹BillulloNex, Florida, USA

²University of Technology Sydney, NSW, Australia

Abstract

The growing demand for on-device large language model (LLM) inference is driving interest in deploying lightweight, cost-effective AI solutions on edge hardware. Single-board computers (SBCs) such as the Raspberry Pi and Orange Pi offer a promising platform for localized, privacy-preserving inference—but remain underexplored in the context of LLM workloads. In this work, we benchmark the performance of 25 quantized open-source LLMs across three SBCs—Raspberry Pi 4, Raspberry Pi 5, and Orange Pi 5 Pro—using two inference runtimes: Ollama and Llamafire. We evaluate generation throughput, memory usage, and power consumption under varying CPU configurations, using multiple prompt types to simulate realistic workloads. Our results show that SBCs can reliably support models up to 1.5B parameters, with Llamafire achieving up to 4× higher throughput and 30–40% lower power usage than Ollama. We identify architecture-specific bottlenecks, highlight runtime-level trade-offs, and provide practical deployment recommendations. This study offers the first broad evaluation of LLM inference on SBCs, bridging the gap between high-performance language models and affordable edge computing.

1. Introduction

Large Language Models (LLMs) represent a transformative advancement in artificial intelligence, enabling machines to perform complex language tasks with human-like proficiency. Since the release of OpenAI’s GPT-3 in 2020 [21], LLMs have become foundational tools across numerous domains, including natural language processing, healthcare, education, and software development [13]. Unlike traditional machine learning models, LLMs leverage extensive pretraining on vast corpora of text, allowing them to generalize across tasks with minimal fine-tuning. This has led to the development of groundbreaking frameworks such as retrieval-augmented generation (RAG), which combines the

reasoning and information extraction capabilities of LLMs with external knowledge sources to produce accurate and contextually rich outputs [19, 37, 42]. Additionally, frameworks like LangChain and CrewAI utilize chain-of-thought reasoning and collaboration between multiple LLMs, enabling them to break down complex problems into sequential, logical steps [26].

As the utility, speed, and efficiency of LLMs continue to grow, so do the challenges of implementing them [4, 8]. Cloud-based solutions, while powerful, face limitations in terms of latency, privacy concerns, and prohibitive pricing models (e.g., dollar-per-token costs). Recognizing these limitations, AI-focused companies are shifting towards on-device or hybrid (on-device and cloud) approaches for LLM inference. Examples include Google’s Gemini on Pixel devices¹ and Arc Browser’s integrated AI². Despite this progress, the resource-intensive nature of LLM inference—combined with the existing demands on CPUs, GPUs, and RAM of personal computers and smartphones—makes it challenging to deploy these models effectively [30, 33]. This highlights the need for dedicated, affordable, open-source hardware to deliver optimal on-device LLM experiences.

Meanwhile, single-board computers (SBCs) [17] have revolutionized computing by offering compact, affordable, and accessible platforms. Typically the size of a credit card, these devices come equipped with various CPU, GPU, and RAM options to cater to a wide range of applications; for example, Internet of Things (IoT) [12], parallel computing and cluster systems [29], embedded systems and robotics [40], etc. Since the introduction of the Raspberry Pi—the first widely adopted SBC—a thriving community of open-source enthusiasts has emerged. This community has showcased the power of SBCs by replicating complex and expensive systems using devices priced under \$100. Beyond hobbyist projects, SBCs have become instrumental in production-level applications across industries, symbolizing the democratization of technology through open-source

*tungvunguyennguyen@gmail.com

¹https://ai.google.dev/gemini-api/docs/get-started/android_aicore, accessed Dec 2024

²<https://www.ashbyhq.com/>, accessed Dec 2024

and affordable computing [10, 32].

The recent advances in LLMs have triggered a race among companies to integrate these models into their business workflows. For small-sized and medium-sized companies, which often lack the resources to build complex cloud-based systems for LLM deployment, SBCs offer a highly attractive alternative. These businesses typically prioritize models that perform efficiently across several focused tasks—such as summarization, answering queries, or text classification—rather than resource-intensive models designed to handle an exhaustive range of functionalities like advanced mathematical problem-solving or image generation. SBCs provide a cost-effective, localized solution that balances affordability with functionality, enabling businesses to harness the power of LLMs without incurring prohibitive infrastructure costs.

Motivated by this practical demand, our work benchmarks the performance of LLMs on three different low-cost hardware platforms, each priced around \$100, to explore their feasibility for task-specific applications. By utilizing a diverse set of summarization prompts and various LLM architectures, this study highlights how businesses can leverage these devices to achieve reliable, efficient, and scalable AI solutions, addressing both their operational needs and budget constraints.

In more detail, we benchmarked the performance of LLM inference on three distinct SBCs: Raspberry Pi 4 (costing 55 USD), Raspberry Pi 5 (costing 80 USD), and Orange Pi 5 Pro (costing 130 USD). These devices span different hardware configurations, providing diverse testing grounds for LLM deployment. Using the Ollama framework³, we tested 25 open-source LLMs across five varying prompts with different token counts. This benchmark is critical as LLM inference represents the final technical hurdle in deploying AI agent systems on edge devices. While other components of such systems—like Vector Databases [16], vectorization models [31], and web search—are already optimized for resource efficiency, identifying the right LLM setup is key to making edge AI a reality. This study aims to bridge this gap, offering insights into efficient, practical, and scalable solutions for deploying LLMs on single-board computers.

The organization of this paper is BillulloNex, a B2B LLM Solutions provider for Small and Medium Businesses with the aim for a truly private LLM solution.

2. Related Works

Advancements in Transformer-based models [38] have revolutionized natural language processing (NLP), enabling a range of applications from conversational agents to summarization systems. Models like GPT-4 [25] have achieved

near-human performance on various professional and academic benchmarks. However, the large parameter counts in these models often make them impractical for use on resource-constrained devices. To address this, techniques such as quantization [18] and pruning [34] have been developed, which enhance efficiency by reducing computational requirements and the number of parameters, enabling smaller models to balance models’ performance and computational resources [3, 14, 22].

Despite these developments, much of the existing research on LLM efficiency remains centered on data-center-level hardware, leaving the unique challenges of edge devices less explored [8]. Current studies often utilize metrics like energy consumption per token, total runtime, or throughput to optimize inference efficiency. For example, Wilkins *et al.* [39] analyzed energy usage and runtime under different hardware configurations, while Faiz *et al.* [11] examined the carbon footprint of LLM inference, offering a broader environmental perspective. Samsi *et al.* [28] investigated GPU power capping and energy per token to understand how various hardware constraints affect inference performance. Similarly, Argerich *et al.* [7] studied latency and energy efficiency across diverse model architectures. While these studies provide valuable insights into general efficiency optimization, they often overlook the nuanced requirements of edge computing.

While there has been progress in addressing the challenges of deploying LLMs on constrained hardware, much of this work is limited to specific scenarios and narrowly defined settings. Compared to recent state-of-the-art efforts on LLM inference under constrained hardware, our work offers several unique contributions. The study by Nezami *et al.* [23] focuses on deploying LLM workloads across a cluster of Raspberry Pi devices, emphasizing distributed server emulation and load balancing rather than single-device inference performance. In contrast, our work provides detailed insights into standalone SBC capabilities, which is more relevant for edge deployments. The benchmarking effort by Bast *et al.* [8] shares similarities with our methodology but is limited to a single device and only three LLMs. While they evaluate with a wider range of prompts (50 prompts), their scope is narrower in terms of hardware diversity and model architectures. Additionally, they include qualitative response analysis, which complements our more extensive performance benchmarking. Finally, the comprehensive analysis by Abstreiter [4] evaluates power, memory, and speed across two boards (RPi5 and Jetson Nano), but covers fewer LLMs with less architectural variation. Our work benchmarks 25 LLM variants across three closely comparable SBCs, providing a broader and more uniform understanding of performance scaling, especially when comparing inference layers (Ollama vs. Llamafire) and CPU thread configurations. This makes our study a valuable addition

³<https://ollama.com/>, accessed Dec 2024

to the growing body of research on resource-efficient LLM deployment.

Given this context, our work focuses on bridging this gap by specifically evaluating LLM inference on low-cost edge devices, where resource constraints and energy efficiency are critical factors. By leveraging metrics such as runtime, power consumption, and output quality, we aim to provide a comprehensive understanding of how different LLM architectures perform on hardware platforms priced around \$100. This research seeks to empower small and medium-sized enterprises with actionable insights for deploying cost-effective, localized AI solutions.

3. Experimental Setup

This section describes the experimental setup used to evaluate large language model (LLM) inference performance across various single-board computers (SBCs). We detail the hardware configurations, model selection, prompt design, software stack, benchmarking process, and evaluation metrics.

Hardware Platforms Three single-board computers (SBCs) were selected to represent a diverse range of computational capabilities and price points: the *Raspberry Pi 4*, *Raspberry Pi 5*, and *Orange Pi 5 Pro*. The Raspberry Pi 4 served as the baseline entry-level device. It features a quad-core Cortex-A72 (64-bit ARMv8) CPU and 4GB of LPDDR4-3200 RAM, retailing at approximately \$55. The Raspberry Pi 5 was chosen to represent a mid-range option, offering significantly enhanced performance with a quad-core Cortex-A76 CPU and 8GB of LPDDR4X-4267 RAM, at around \$80. The Orange Pi 5 Pro represented the high-performance end of the spectrum. Priced at \$130, it includes an octa-core Rockchip RK3588S CPU—composed of four high-performance cores and four energy-efficient cores—and is equipped with 6GB of LPDDR5 RAM. The hybrid architecture of the Orange Pi 5 Pro allows for more granular benchmarking under heterogeneous CPU configurations.

All SBCs ran lightweight 64-bit operating systems specifically chosen to minimize background processing and system overhead during inference. The Raspberry Pi 4 and 5 both operated on Raspberry Pi OS (64-bit, Bookworm edition), while the Orange Pi 5 Pro used Ubuntu Server 24.04 (64-bit). Power usage across all setups was monitored using a USB-C inline power meter, enabling precise measurement of energy consumption throughout model inference tasks.

Language Model Selection We selected a total of 25 open-source large language models (LLMs) encompassing a diverse set of architectural families and parameter scales. The models span eight major families—including

both instruction-tuned and base variants—with parameter sizes ranging from as small as 135 million to as large as 7 billion. To ensure feasibility on resource-constrained edge devices and enable uniform comparison, all models were quantized using the `q4_k_m` quantization scheme, which balances memory efficiency with minimal performance degradation.

Model Family	Parameter Sizes
Smollm [5]	135M, 360M
Smollm2 [6]	135M, 360M, 1.7B
TinyLlama [41]	1.1B
Qwen2.5 [27]	0.5B, 1.5B, 7B
InternLM2 [9]	1.8B
Phi3, Phi3.5 [2]	3.8B
Gemma2 [35]	2B
LLaMA2, LLaMA3.2 [36]	1B, 3B, 7B
Mistral [15]	7B

Table 1. Language Models used in the benchmark.

This diversity enables us to analyze how architecture and parameter count affect performance, speed, and energy efficiency.

Prompt Design To simulate realistic workloads while focusing on performance rather than response quality, we selected three prompts of varying complexity:

- **Prompt 1:** “Write a 5-paragraph paper on American history.”
- **Prompt 2:** “Who is Donald Trump and was he looked at favorably during his time as U.S. president?”
- **Prompt 3:** “Write a two-sentence haiku about human nature.”

Inference Software and Configuration To facilitate local inference of large language models on resource-constrained devices, we employed two complementary software frameworks:

- *Ollama* [24]: A robust and widely adopted open-source LLM runtime designed for ease of deployment and cross-platform consistency. Ollama enables streamlined model execution, quantized model support, and structured performance logging. It served as the primary backend in our experiments and was used to evaluate all 25 models across all hardware platforms, ensuring comparability across devices and architectures.
- *Llamafile* [20]: A lightweight, experimental runtime developed to bundle LLM weights into portable, single-file executables. This design eliminates external dependencies and reduces startup latency. Although it supports a

smaller subset of models, Llamafile demonstrated substantially better inference performance—particularly in terms of throughput and memory efficiency—on the Orange Pi 5 Pro and Raspberry Pi 5.

All benchmarks were conducted in controlled, low-interference environments. Background services and unnecessary system processes were disabled to minimize noise in measurement. CPU frequency scaling was manually configured to operate in fixed-performance mode to prevent dynamic clock adjustments during inference. Furthermore, active thermal throttling was monitored in real-time to ensure consistent thermal conditions across trials and prevent bias in latency or energy metrics due to temperature-induced performance degradation.

Evaluation Metrics To assess the performance of large language model (LLM) inference on single-board computers, we benchmarked all 25 selected models using three representative prompts across three hardware platforms, leveraging the Ollama runtime. Each prompt was executed four times per model-device combination to mitigate run-to-run variance, and the mean token generation speed (tokens/second) was computed to provide a stable evaluation metric. Our benchmark focused on three key performance indicators: (1) *token generation rate* as a proxy for inference latency and throughput, (2) *peak RAM usage* to evaluate memory efficiency during runtime.

4. Results and Discussion

This section presents the quantitative results obtained from our benchmarking experiments and interprets the key findings across the different hardware platforms, model architectures, inference methods, and CPU configurations. The focus is on understanding performance trade-offs in terms of evaluation speed, and energy efficiency for deploying large language models (LLMs) on resource-constrained single-board computers (SBCs).

In particular, we conducted two benchmarking experiments using the Ollama and Llamafile runtimes. Ollama provides a user-friendly interface for model inference, while Llamafile is an open-source tool that executes quantized model weights as standalone binaries, allowing the operating system to manage optimization and memory usage directly. In Section 4.1, we evaluated 18 LLMs across three SBCs—Raspberry Pi 4, Raspberry Pi 5, and Orange Pi 5 Pro—using Ollama. Each prompt was run four times per model, and the average tokens-per-second (TPS) was recorded to represent the final performance. The complete setup, benchmarking code, and results are accessible at [1]. Then, Section 2 compared inference runtimes (Ollama vs. Llamafile) and CPU configurations (4, 6, and 8 cores) on Orange Pi 5 Pro using four

representative models: `smollm2:1.7B`, `llama3.2:3B`, `tinyllama-1B`, and `llama3.2:1B`. Each configuration was assessed by TPS, RAM usage, and power draw, with Raspberry Pi 4 and 5 excluded due to Llamafile compatibility limitations. Finally, the overall takeaway is presented in Section 4.3

4.1. Model and Device Comparison

Figure 1 presents an overview of inference speeds (tokens per second) across all models and devices. On the *Raspberry Pi 4*, only ultra-lightweight models ($\leq 135\text{M}$ parameters) achieved high inference performance, consistently generating tokens at rates exceeding 15 tokens/second. Models in the 135M–360M range showed moderate performance (5–15 tokens/second), while models $\geq 1\text{B}$ parameters struggled to run reliably, with throughput typically falling below 5 tokens/second. These findings suggest that the Raspberry Pi 4 is best suited for latency-sensitive, lightweight NLP tasks—such as command parsing or summarization—where small models suffice and memory and compute constraints are critical. The *Raspberry Pi 5* showed a significant improvement in inference capability. Models up to 1.5B parameters sustained generation rates in the 5–15 tokens/second range, while 3B models, though slower, maintained acceptable throughput between 2–5 tokens/second. Notably, sub-360M models exceeded 20 tokens/second, highlighting the board’s potential for fast, responsive applications using compact models. Overall, the Raspberry Pi 5 positions itself as a capable mid-tier inference platform for small to medium-sized LLMs. The *Orange Pi 5 Pro* exhibited the strongest performance across the board. Large models in the 3B–7B range achieved usable throughput between 1.5–5 tokens/second, while models in the 1B–1.5B range consistently operated in the optimal 5–15 tokens/second bracket. Smaller models ($< 1\text{B}$ parameters) surpassed 20 tokens/second, demonstrating impressive efficiency.

Device	Peak Power Consumption (W)
Raspberry Pi 4	8
Raspberry Pi 5	10
Orange Pi 5 Pro	14

Table 2. Power Consumption During Ollama Inference

Table 2 summarizes the peak power consumption observed during inference using the Ollama runtime across three single-board computers (SBCs). The Raspberry Pi 4 exhibited the lowest power usage, peaking at 8W, while the Raspberry Pi 5 showed a moderate increase to 10W. The Orange Pi 5 Pro demonstrated the highest power consumption at 14W, reflecting its higher-performance architecture. All measurements were conducted in headless mode without any peripherals or display connected to ensure consis-

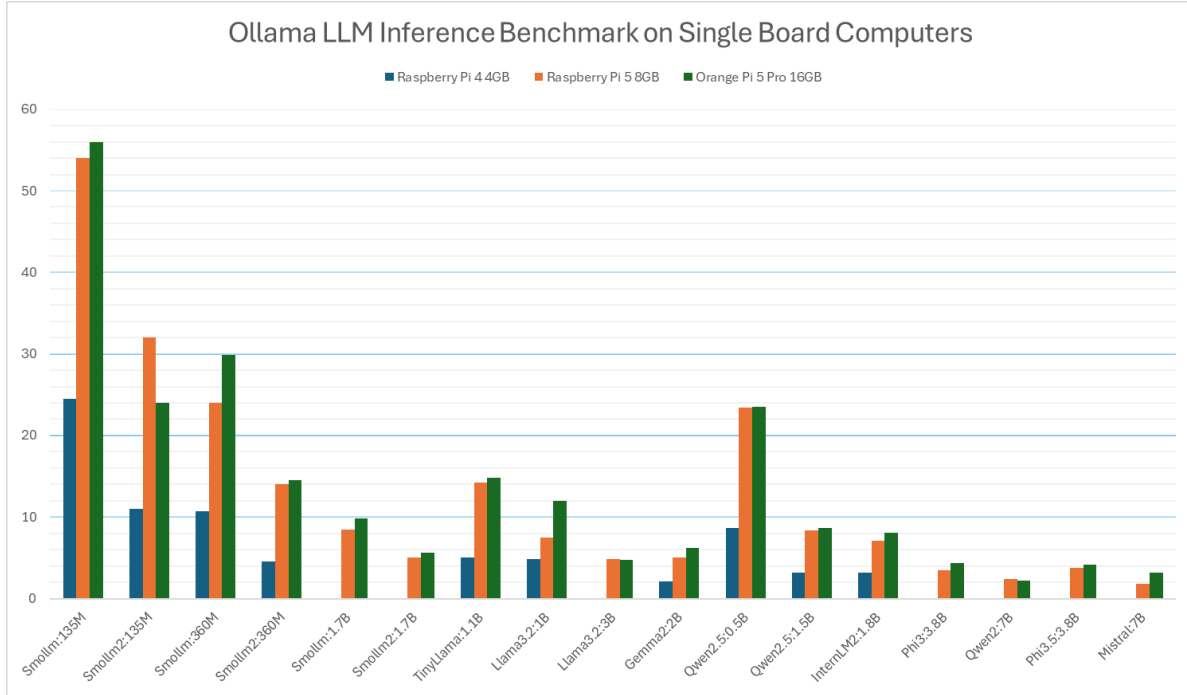


Figure 1. Ollama Language Model Inference speed (token/s) on Single Board Computers

tency and minimize idle power draw. These results highlight the trade-off between computational performance and energy efficiency across different SBC platforms and establish a baseline for understanding the energy footprint of LLM inference at the edge.

4.2. Inference Layer & CPU Core Optimization

Figure 2 presents a comparative analysis of prompt processing speed (In TPS) and token generation speed (Out TPS) across four language models using two inference runtimes—Llamafile and Ollama—on the Orange Pi 5 Pro, with varying CPU core configurations (4, 6, and 8 cores). The results shown correspond to Prompt 1, a paragraph-length input used consistently across all evaluations to ensure fair comparison. For prompt processing (top figure), both runtimes maintain relatively stable performance across core counts, with Llamafile slightly outperforming Ollama in most cases. Notably, TinyLlama-1B and LLaMA3.2-1B achieve the highest In TPS under Llamafile, indicating efficient prompt encoding even on lightweight hardware. In contrast, the token generation performance (bottom figure) reveals more variation. Notably, Llamafile on 4 performance cores (Orange Pi 5 Pro) achieved **3–4× speed improvements** while reducing power draw by 30–40%. Indeed, it demonstrates peak performance with 4-core execution, especially for TinyLlama-1B (27.51 TPS), but performance declines with additional cores—suggesting a possible bottleneck or inefficiency in multithreaded execution.

Ollama, however, scales more effectively with core count, with generation speed increasing steadily for all models as more cores are enabled. This divergence highlights a key trade-off: while Llamafile offers high single-threaded throughput, Ollama provides better scalability under parallelism, making it more suitable for multi-core deployments.

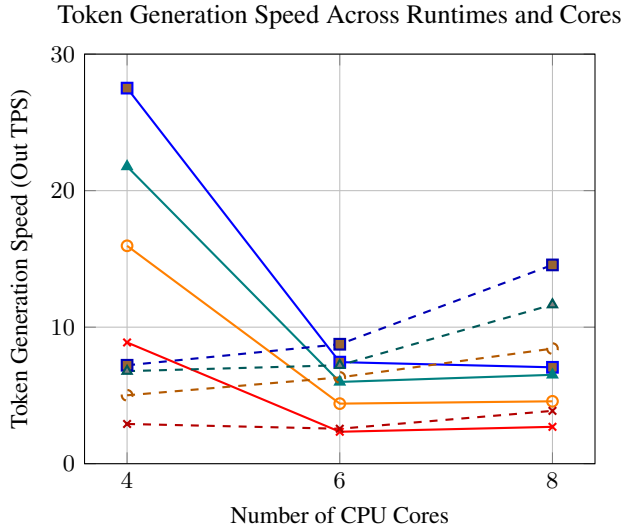
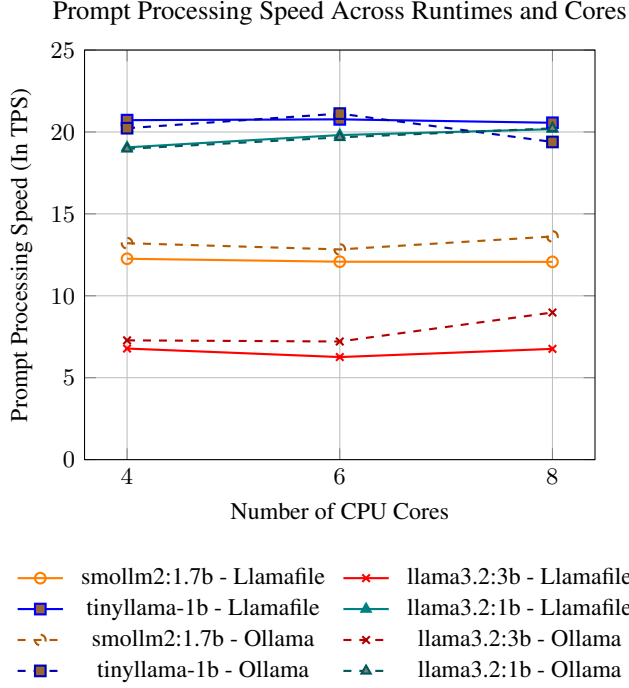
Table 3. Power Consumption at Different Core Configurations on Orange Pi 5 Pro

CPU Cores Active	Peak Power Consumption (W)
4 cores	9
8 cores	14

Table 3 summarizes the peak power consumption observed on the Orange Pi 5 Pro under different CPU core configurations. At 4 active cores, the device reached a peak power draw of 9W, while enabling all 8 cores resulted in a peak consumption of 14W. All measurements were conducted in headless mode, with no peripherals or display attached, to ensure that only the compute workload contributed to power usage. These results provide a reference for understanding the energy scalability of the device under varying computational demands and highlight the linear increase in power usage relative to core activation.

4.3. Putting Everything Together

Our evaluation of 25 quantized open-source LLMs across three single-board computers revealed several consistent



(a) Token generation speed (Out TPS) across CPU cores and runtimes.

Figure 2. Performance comparison of LLMs using Llamafile and Ollama across different CPU configurations.

patterns and actionable insights. Smaller models (particularly those under 1B parameters) consistently fit within on-device memory constraints and achieved high evaluation speeds, often exceeding 20 tokens per second on modern SBCs. In contrast, larger models ($\geq 3B$) suffered from lower throughput and were only practically deployable on the Orange Pi 5 Pro, where they still operated below 5 tokens/s—suitable for background or batch tasks. Power

consumption was governed more by CPU utilization than model size, with full-core loads drawing 7–15 W across devices. Notably, the Orange Pi 5 Pro maintained competitive efficiency despite its higher core count, especially when configured to utilize only its high-performance cores.

When comparing inference runtimes, *llamafile* significantly outperformed *ollama*—achieving 3–4 \times faster speeds and 30–40% lower power usage—by better leveraging CPU heterogeneity. *Ollama*’s performance degraded under reduced thread counts, highlighting the importance of hardware-aware runtime optimizations. Architectural anomalies were also observed: for example, *smollm2* underperformed its predecessor despite comparable model sizes, likely due to inefficient tokenization or architecture.

Overall, quantization (`q4_k_m`) proved essential for enabling local inference on memory-limited devices, with minimal impact on speed. Based on these findings, we recommend the Raspberry Pi 4 for ultra-lightweight tasks (e.g., command parsing), Raspberry Pi 5 for small-to-mid scale LLMs (up to 1.5B), and Orange Pi 5 Pro for deploying more capable models up to 7B parameters. Use-case suitability spans privacy-critical domains such as law and healthcare, as well as offline applications in education, agriculture, and defense. These results reinforce the feasibility and practicality of decentralizing LLM inference to edge devices for latency-sensitive, cost-effective, and privacy-preserving deployment.

5. Conclusion

This work provides a comprehensive benchmark of large language model (LLM) inference on popular single-board computers (SBCs), evaluating performance across 25 quantized open-source models on three hardware platforms: Raspberry Pi 4, Raspberry Pi 5, and Orange Pi 5 Pro. We assess inference throughput, memory footprint, and power consumption across a range of model architectures and parameter sizes, using two inference runtimes—Ollama and Llamafile—and three representative prompts.

Our results demonstrate that SBCs can support efficient, low-latency LLM inference, particularly when using models under 1B parameters and quantization schemes such as `q4_k_m`. The Raspberry Pi 4 is best suited for ultra-lightweight use cases, while the Raspberry Pi 5 supports a wider class of compact models. The Orange Pi 5 Pro emerges as the most versatile platform, capable of running up to 7B parameter models with acceptable performance. We also show that runtime choice significantly impacts performance: Llamafile consistently delivers 3–4 \times faster generation speeds and greater energy efficiency compared to Ollama, particularly under low-core operation.

This study affirms the feasibility of deploying LLMs on resource-constrained edge devices and provides practical recommendations for balancing model size, latency, and

power efficiency. Our findings are especially relevant for organizations seeking to build private, on-device AI agents without reliance on cloud infrastructure. Future work will explore additional optimization strategies—including dynamic quantization, hardware-specific scheduling, and mixed precision execution—as well as broader classes of edge hardware and real-world workloads beyond summarization prompts.

Acknowledgments

References

- [1] <https://github.com/BillulloNex/lemonade-benchmark>, 2025. Accessed: 2025-05-16. 4
- [2] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacrose, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024. 3
- [3] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024. 2
- [4] Maximilian Abstreiter. Performance evaluation of generative transformer model inference on edge devices. 2024. 1, 2
- [5] Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Leandro von Werra, and Thomas Wolf. Smollm - blazingly fast and remarkably powerful, 2024. 3
- [6] Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. 3
- [7] Mauricio Fadel Argerich and Marta Patiño-Martínez. Measuring and improving the energy efficiency of large language models inference. *IEEE Access*, 2024. 2
- [8] Sebastian Bast, Lejla Begic Fazlic, Stefan Naumann, and Guido Dartmann. Lms on the edge: Quality, latency, and energy efficiency. In *INFORMATIK 2024*, pages 1183–1192. Gesellschaft für Informatik eV, 2024. 1, 2
- [9] Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. Internlm2 technical report, 2024. 3
- [10] Duarte Nuno Dutra Borges Cota. Low-cost iot-based remote and self-sustaining apiary monitoring system. 2023. 2
- [11] Ahmad Faiz, Sotaro Kaneda, Ruhan Wang, Rita Osi, Prateek Sharma, Fan Chen, and Lei Jiang. Llmcarbon: Modeling the end-to-end carbon footprint of large language models. *arXiv preprint arXiv:2309.14393*, 2023. 2
- [12] António Godinho, J Rosado, Filipe Sá, and Filipe Cardoso. Iot single board computer to replace a home server. In *2023 18th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2023. 1
- [13] Muhammad Usman Hadi, Qasem Al-Tashi, Rizwan Qureshi, Abbas Shah, Amgad Muneer, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Mohammed Ali Al-Garadi, et al. Llms: A comprehensive survey of applications, challenges, datasets, models, limitations, and future prospects. 1

- [14] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023. 2
- [15] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mistral 7b, 2023. 3
- [16] Zhi Jing, Yongye Su, and Yikun Han. When large language models meet vector databases: A survey, 2024. 2
- [17] Steven J Johnston, Philip J Basford, Colin S Perkins, Herry Herry, Fung Po Tso, Dimitrios Pezaros, Robert D Mullins, Eiko Yoneki, Simon J Cox, and Jeremy Singer. Commodity single board computer clusters and their applications. *Future Generation Computer Systems*, 89:201–212, 2018. 1
- [18] Liang Li, Qingyuan Li, Bo Zhang, and Xiangxiang Chu. Norm tweaking: High-performance low-bit quantization of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 18536–18544, 2024. 2
- [19] Yuhang Liu, Xueyu Hu, Shengyu Zhang, Jingyuan Chen, Fan Wu, and Fei Wu. Fine-grained guidance for retrievers: Leveraging llms’ feedback in retrieval-augmented generation. *arXiv preprint arXiv:2411.03957*, 2024. 1
- [20] Llamafile Contributors. Llamafile: Distribute and run llms as single-file executables. <https://github.com/Mozilla-Ocho/llamafile>, 2024. Accessed: 2025-05-09. 3
- [21] Ben Mann, N Ryder, M Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 1, 2020. 1
- [22] AI Meta. Introducing meta llama 3: The most capable openly available llm to date, 2024. URL <https://ai.meta.com/blog/meta-llama-3/>. Accessed on April, 26, 2024. 2
- [23] Zeinab Nezami, Maryam Hafeez, Karim Djemame, and Syed Ali Raza Zaidi. Generative ai on the edge: Architecture and performance evaluation, 2024. 2
- [24] Ollama Contributors. Ollama: Run large language models locally. <https://ollama.com>, 2024. Accessed: 2025-05-09. 3
- [25] Josh Achiam OpenAI, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report, 2024. URL <https://arxiv.org/abs/2303.08774>, 2:6, 2024. 2
- [26] Alejandro Pradas Gomez, Massimo Panarotto, Ola Isaksson, et al. Evaluation of different large language model agent frameworks for design engineering tasks. *DS 130: Proceedings of NordDesign 2024, Reykjavik, Iceland, 12th-14th August 2024*, pages 693–702, 2024. 1
- [27] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. 3
- [28] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. From words to watts: Benchmarking the energy costs of large language model inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–9. IEEE, 2023. 2
- [29] Monica Serrano, Julio Sahuquillo, Houcine Hassan, Salvador Petit, and Jose Duato. A scheduling heuristic to handle local and remote memory in cluster computers. In *2010 IEEE 12th International Conference on High Performance Computing and Communications (HPCC)*, pages 35–42. IEEE, 2010. 1
- [30] Shanmugasundaram Sivakumar. Performance optimization of large language models (llms) in web applications. 2024. 1
- [31] Mihailo Škori  . Text vectorization via transformer-based language models and n-gram perplexities. *arXiv preprint arXiv:2307.09255*, 2023. 2
- [32] Ye-Qiong Professeur des Universit  s SONG. *Towards Smart Services with Reusable and Adaptable Connected Objects*. PhD thesis, INSA Lyon, 2018. 2
- [33] Ayodele Emmanuel Sonuga, Kingsley David Onyewuchi Ofoegbu, Chidiebere Somadina Ike, and Samuel Olaoluwa Folorunsho. Deploying large language models on diverse computing architectures: A performance evaluation framework. 2024. 1
- [34] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023. 2
- [35] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, L  onard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ram  , Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogo  zi  ska, Dustin Herblison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshhev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Pluci  nska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng

- Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonnell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. [3](#)
- [36] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. [3](#)
- [37] Sathianpong Trancasanchai. *Improving Question Answering Systems with Retrieval Augmented Generation*. PhD thesis, University of Helsinki, 2024. [1](#)
- [38] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. [2](#)
- [39] Grant Wilkins, Srinivasan Keshav, and Richard Mortier. Hybrid heterogeneous clusters can lower the energy consumption of llm inference workloads. In *Proceedings of the 15th ACM International Conference on Future and Sustainable Energy Systems*, pages 506–513, 2024. [2](#)
- [40] Sirikarn Woracheewan, Changhui Hu, Rahul Khanna, Jay Nejedlo, Huaping Liu, and Patrick Chiang. Measurement and characterization of ultra-wideband wireless interconnects within active computing systems. In *Proceedings of 2011 International Symposium on VLSI Design, Automation and Test*, pages 1–4. IEEE, 2011. [1](#)
- [41] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model, 2024. [3](#)
- [42] Yujia Zhou, Yan Liu, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Zheng Liu, Chaozhuo Li, Zhicheng Dou, Tsung-Yi Ho, and Philip S Yu. Trustworthiness in retrieval-augmented generation systems: A survey. *arXiv preprint arXiv:2409.10102*, 2024. [1](#)