CSCI-582 Project Report: Comparative Analysis of Edge Deployment of LLaMa Models on NVIDIA's Jetson AGX Orin

Nathaniel Graves
Colorado School of Mines
Golden, CO, USA

Joline Sikora
Colorado School of Mines
Golden, CO, USA

1 INTRODUCTION

With the increasing complexity of tasks being handled by large language models (LLMs), deploying them efficiently on edge devices like NVIDIA's Jetson AGX Orin is essential. This paper compares the performance of two advanced models: Sheared-LLaMa-2.7B and Open_LLaMa_3B_v2, focusing specifically on their operational viability in constrained environments. The efficiency of these deployments is critical due to the intensive computational demands and real-time processing requirements of modern applications. By evaluating the performance of these models on the NVIDIA Jetson AGX Orin, this study aims to highlight their suitability for edge computing, providing insights into the trade-offs between computational power and operational efficiency in real-world scenarios. GPT-2 is employed as a baseline to provide a comparative benchmark, illustrating how advancements in model design and implementation can enhance deployment on edge devices.

2 TARGETED DOMAIN [2 PTS]

This research investigates the deployment of LLMs within edge computing frameworks. Edge computing architectures are designed to process data close to the source of generation, significantly reducing latency and bandwidth usage. This is crucial for applications requiring immediate computational responses, such as IoT devices and autonomous systems, where rapid processing can critically impact performance and functionality.

3 MOTIVATION: THE NEED FOR ACCELERATION [2 PTS]

Recent advancements in hardware architectures have increasingly emphasized the utility of specialized accelerators such as GPUs, TPUs, and FPGAs over traditional CPU-based computing for tasks that demand high computational throughput and energy efficiency. This transition is particularly notable in applications involving deep learning and LLMs, where the computational and power limitations of CPUs are significant constraints.

In edge computing contexts, the acceleration of computational processes is critical. By processing data directly at the source rather than relying on cloud-based services, edge computing reduces latency and bandwidth. These characteristics are essential for real-time applications, such as those found in autonomous vehicles and interactive AI systems, where rapid data processing is necessary for effective operation.

The deployment of LLMs such as Sheared Llama[7] and Open Llama[1] on specialized hardware platforms like the Nvidia Jetson AGX Orin enables enhanced processing capabilities at the edge. A local setup not only mitigates the latency and bandwidth issues associated with centralized computing but also leverages the computational power and efficiency of advanced hardware accelerators. Assessing the performance of these models on such platforms is vital for understanding their practical effects in edge-based applications. Using the GPT-2 baseline[5] allows us to gauge the improvements these advanced models offer over more traditional approaches, providing critical insights into their deployment and operational efficiency in real-world scenarios.

4 RELATED WORK [8 PTS]

The optimization of neural networks for deployment on low-power and resource-constrained devices has been a significant focus of existing research. Many studies have explored general model optimization techniques, but specific investigations into the challenges of deploying contemporary LLMs such as Sheared_Lllama[7] and Open_Llama[1] on edge devices are relatively scarce. This paper builds upon the foundational work presented in recent literature, including the innovative approaches developed by Wang et al. (2023) in "Tabi: An Efficient Multi-Level Inference System for Large Language Models," which introduces a multi-level inference system to optimize inference latency and accuracy [6]. The "Tabi" system combines real-time data processing with advanced model management to facilitate the deployment of LLMs on edge devices. It features a decision-making engine that assesses query complexity in real-time and determines the most suitable processing tier. This dynamic model selection conserves time and resources, and the system's integration with existing edge computing architectures improves scalability and responsiveness.

The "Tabi" system also incorporates advanced techniques such as calibrated confidence measurement and context-aware processing, which help to optimize the decision-making process regarding when to escalate queries from smaller models to full LLMs. These features ensure that the system maintains high accuracy while keeping computational load at a manageable level, making it ideal for deployment in environments where resources are limited but demand for intelligent processing is high.

Previous works by Han et al. (2015) and others have demonstrated the effectiveness of model compression techniques like pruning and quantization [2]. These techniques are essential for reducing the computational demands of neural networks on edge devices. Additionally, their integration with knowledge distillation in methods like PQK provides a robust framework for enhancing model efficiency, which is critical for the real-time processing needs of edge-deployed LLMs [3]. Moreover, Ma et al. (2018) explore sparse learning methods that optimize network architectures specifically for edge computing, offering Open_LLaMa into architectural adjustments that further improve operational efficiency [4].

Together, these studies form a comprehensive backdrop for our research, underscoring the importance of model optimization and

resource management for the effective deployment of LLMs on edge devices.

4.1 Current Limitations

Despite significant advancements in deploying LLMs on edge devices, several limitations persist that can impact their effectiveness and scalability:

Power Consumption: While models like Sheared-LLaMa-2.7B[7] and Open_LLaMa_3B_v2[1] are optimized for efficiency, the power consumption required to operate such advanced models on edge devices still poses a challenge. The energy demands can limit their deployment in scenarios where power availability is constrained or where long battery life is critical.

Model Complexity: Although efforts like pruning and quantization have made strides in reducing model size and complexity, managing these models on edge devices with limited computational resources remains a substantial challenge. The balance between model complexity and performance is not always optimal, which can lead to decreased accuracy or increased latency in real-world applications.

Latency: For real-time applications, such as those required in autonomous driving or direct communication, even minimal latency can be detrimental. The process of optimizing LLMs to reduce latency without compromising performance is complex and often has significant issues.

5 PROJECT DESCRIPTION [2 PTS]

This project focuses on deploying and optimizing advanced large language models, specifically Sheared-LLaMa-2.7B and Open_LLaMa_3B_v2, for real-time natural language processing (NLP) applications on edge devices. The benchmarks are designed to evaluate the performance of these models in tasks that demand rapid processing and high accuracy. Both Sheared-LLaMa and Open_LLaMa are quantized to 16 bits to enhance computational efficiency on the NVIDIA Jetson AGX Orin platform, with parameter counts of 2.7 billion and 3 billion, respectively. GPT-2, used as a baseline for comparison, is also quantized to 16 bits and has approximately 1.5 billion parameters, providing a benchmark to evaluate the advancements in model design and implementation.

Selected Application/Benchmark: The benchmarks involve real-time translation and interactive communication tasks to assess the models' performance in scenarios requiring low-latency and high-throughput processing. Data was collected using NVIDIA's 'tegrastats' tool, which provides real-time statistics on GPU and CPU usage, memory consumption, and power usage. This tool enabled continuous monitoring of system resources throughout the duration of model operation, ensuring accurate capture of performance impact and providing quantitative data on the operational efficiency of each model under the test conditions.

We ran the benchmarks using the Text Generation Benchmark scripts provided with the HuggingFace Transformers library. This benchmark measured performance for generating 128 new output tokens with 16 bit quantization. The scripts ran each model 3 times. We were given the average run time (in seconds) and the average tokens/second for each run and average runtime, average tokens/second and memory consumption for all 3 runs. See sample output below:

```
WARMUP 0 = 16.0266 seconds, 8.0 tokens/sec
WARMUP 1 = 12.9703 seconds, 9.9 tokens/sec
RUN 2 = 12.9649 seconds, 9.9 tokens/sec
RUN 3 = 12.9685 seconds, 9.9 tokens/sec
AVG = 12.9667 seconds, 9.9 tokens/sec  memory=21551.48 MB
```

Targeted Platform and Accelerators: The NVIDIA Jetson AGX Orin, equipped with 2048 CUDA Cores and 64 Tensor Cores, serves as the hardware platform running Linux for Tegra (L4T) r35.3.1 with 29 GiB of RAM.

Programming Interfaces and Runtime Framework: The project utilizes CUDA to maximize model performance on NVIDIA hardware, enabling direct control over GPU resources for optimized computation.

5.1 Challenges

Here are the key challenges faced and the methods used to address them:

Permissions Issues with Docker: Running Docker required sudo access permissions. We resolved the issue by requesting that our user accounts be granted those permissions.

Resource Conflicts Between Models: We encountered issues with two models attempting to access the same autotag temporary file due to the use of a shared '/tmp' directory. This conflict prevented more than one user running models for benchmarking. A workaround was to delete the autotag file between simulations.

Limitations of Pre-trained Models: Working with pre-trained models such as Sheared-LLaMa and Open_LLaMa imposes certain restrictions, as these models come with predefined architectures and parameters. Critical aspects like batch size, learning rates, and layer configurations are typically preset, which limits our ability to optimize them further for specific tasks or hardware constraints. This restricted the scope of our project, as it prevented us from testing different tuning parameters.

6 EXPERIMENT SETUP [2 PTS]

6.1 Methodology

This study employs a structured experimental approach to assess and compare the performance of two large language models, Open Llama and Sheared Llama, on the NVIDIA Jetson AGX Orin edge computing platform. To ensure comprehensive and replicable results, the methodology is divided into hardware setup, software configuration, and performance measurements.

The experimental procedure involved multiple iterations over a standardized set of tasks designed to simulate real-world applications. To eliminate noise from the results and improve reliability, each model was tested under identical conditions with repeated runs. This approach allowed us to gather consistent data on the models' performance across various computational tasks.

6.1.1 Hardware Setup. The experiments were conducted on the NVIDIA Jetson AGX Orin, leveraging its ARM-based CPU and integrated NVIDIA GPU, optimized for machine learning tasks on edge devices.

6.1.2 Software Configuration. Both LLMs, Open Llama and Sheared Llama, were configured to run in the same software environment to ensure fair comparison. The models were deployed using a unified framework to minimize variations in software overhead.

6.1.3 Performance Measurements. This study evaluates the performance of two advanced large language models that are implemented using the Hugging Face Transformers library. Hugging Face Transformers is an open-source library that provides thousands of pre-trained models for a wide range of natural language processing tasks. It offers a standardized approach to loading, training, and deploying models, making it highly popular for both academic research and industry applications due to its ease of use and broad support for different model architectures.

Princeton-NLP/Sheared-LLaMa-2.7B: This model has been optimized to 2.7 billion parameters, enhancing its efficiency for deployment in resource-constrained environments such as edge devices. It is designed to maintain robust performance capabilities while minimizing computational demands. More information about this model can be found at: https://huggingface.co/princeton-nlp/Sheared-LLaMA-2.7B.

OpenLM-Research/Open_LLaMa_3B_v2: An improved version of the LLaMa model with 3 billion parameters, this model aims to provide an excellent balance between computational efficiency and task performance, suitable for demanding applications that require high processing power. Additional details can be found here: https://huggingface.co/openlm-research/open_llama_3b_v2.

Baseline Model - GPT-2: Employed as the baseline, GPT-2 is a well-established model in the Transformers ecosystem, known for its efficacy across a broad spectrum of NLP tasks. This model provides a standard against which the performance and efficiency of the newer models can be measured, offering insights into advancements in model design and implementation. The specifics of GPT-2 are available at its Hugging Face repository page.

By leveraging the Hugging Face Transformers library, each model benefits from a well-maintained and widely-accepted framework, ensuring compatibility and ease of deployment across various computing environments, including edge devices.

6.1.4 Docker Container Setup. To ensure a consistent and reproducible environment across tests, Docker containers were used. The containers were managed using the 'jetson-inference' toolkit, specifically tailored for the NVIDIA Jetson architecture. This toolkit simplifies the deployment and management of machine learning models on Jetson devices. The Docker setup involved pulling a pre-built container optimized for the JetPack version L4T r35.3.1, which is compatible with the NVIDIA Jetson AGX Orin hardware.

We used the following command to pull a Docker container specifically optimized for our JetPack version, L4T r35.3.1:

```
git clone https://github.com/dusty-nv/jetson-containers
bash jetson-containers/install.sh
```

6.1.5 Experimental Procedure.

Benchmark Setup The benchmark involves running the benchmarking tool provided by Hugging Face, 'huggingface_benchmark.py', which tests the models' performance in natural language processing tasks such as text generation, summarization, and translation. Each model is evaluated under the same hardware and software conditions to ensure consistency:

- **Script Execution**: The script is executed on the NVIDIA Jetson AGX Orin platform to leverage its robust computational capabilities.
- **Model Configurations**: Models are configured with predefined parameters suitable for the edge device, focusing on evaluating both speed and accuracy.
- **Tasks and Metrics**: The script measures inference time, memory usage, and output accuracy, providing a comprehensive view of each model's performance.
- **Performance Data**: Includes time taken for each task, number of tokens processed per second, and total memory consumption.
- **System Monitoring**: Utilizes 'tegrastats' to monitor GPU and CPU utilization, as well as power consumption during the benchmark, to assess the models' efficiency under load.

Analysis This project focuses on deploying and optimizing advanced large language models—specifically, the Sheared-LLaMa and Open_LLaMa—for real-time natural language processing (NLP) applications on edge devices. The benchmarks assess these models, which are configured with 16-bit quantization to enhance computational efficiency. Sheared-LLaMa is optimized to 2.7 billion parameters, and Open_LLaMa to 3 billion parameters, to evaluate their performance in demanding tasks that require rapid processing and high accuracy.

6.2 Metrics Measured and Comparisons Made

We measured several key performance metrics to assess and compare the effectiveness of the models on the NVIDIA Jetson AGX Orin platform.

6.2.1 Performance Metrics. Performance metrics were carefully selected to quantify the efficiency and effectiveness of each model under test. Key metrics included:

- **Inference Latency:** The time taken to complete a single inference cycle, measuring the responsiveness of each model.
- **Power Consumption:** Monitored using the integrated tools in NVIDIA Jetson AGX Orin, to assess the energy efficiency of each model during peak and idle conditions.
- **GPU and CPU Utilization:** Recorded to understand the computational load and resource allocation for each model.
- **Memory Usage:** Evaluated to determine how efficiently each model manages its memory footprint during operation.

6.2.2 Comparisons Made. The comparisons in this study are conducted between Sheared-LLaMa-2.7B and Open_LLaMa_3B_v2, focusing on their performance in terms of inference speed, power efficiency, and computational load. GPT-2 serves as the baseline for these comparisons, providing a foundation to gauge the advancements and efficiencies introduced by the newer models.

By assessing performances, the study aims to highlight the specific improvements in speed and efficiency that Sheared-LLaMa-2.7B and Open_LLaMa_3b_v2 offer over the baseline model. This comparative analysis helps in understanding the trade-offs involved

when deploying these models on edge devices, thus providing insights into their practical implications for edge computing.

7 RESULTS [9 PTS]

Visualizations

Table 1: Memory and Power Consumption

Model	Memory Usage (MB)	Power Consumption (mW)	Total Average Power (mW)
GPT-2	2372.21	Varies	2530.47
OpenLlama	14401.98	Varies	3710.30
ShearedLlama	15851.85	Varies	3167.09

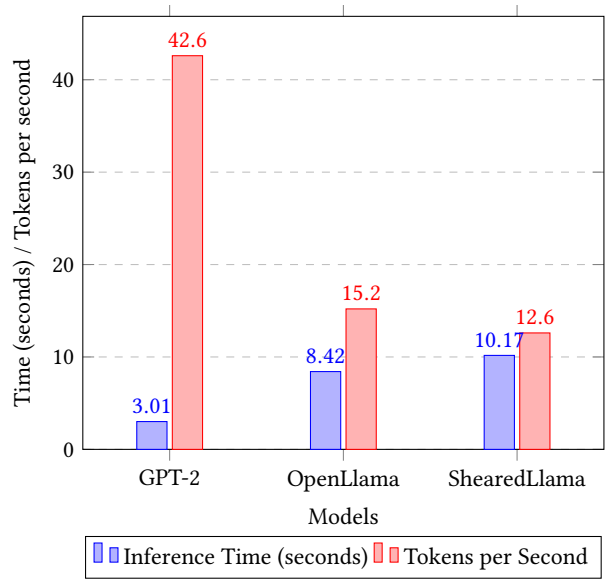

Figure 1: Comparison of Inference Time and Tokens Processed Per Second for Different Models

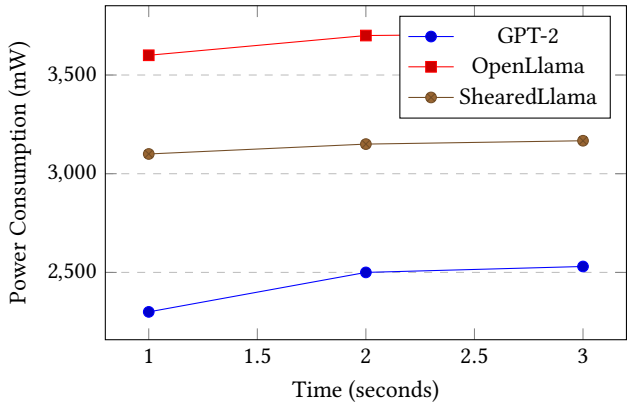

Figure 2: Power Consumption Over Time During Benchmark

7.1 Performance and Power Consumption Analysis

This section discusses the results obtained from the benchmarking of the three models: GPT-2, Open_llama, and Sheared_llama using the `huggingface_benchmark.py` script and `tegrastats` for system monitoring.

7.1.1 Inference Speed and Efficiency. The performance analysis based on inference speed and tokens processed per second is as follows:

- **GPT-2** showed the highest efficiency, completing tasks in an average of **3.0081 seconds** and processing **42.6 tokens per second**.
- **Open_llama** had a moderate performance with **8.4174 seconds** per task and **15.2 tokens per second**.
- **Sheared_llama** was the slowest, taking **10.1690 seconds** per task with a rate of **12.6 tokens per second**.

7.1.2 Memory Utilization. Memory usage during the tasks was recorded as follows:

- **GPT-2** was the most memory-efficient model using **2372.21 MB**.
- **Open_llama** and **Sheared_llama** required **14401.98 MB** and **15851.85 MB**, respectively, indicating a higher demand for resources.

7.1.3 Power Consumption. The power consumption metrics indicate:

- **GPT-2** consumed the least power with **2530.47 mW**.
- **Open_llama** and **Sheared_llama** consumed **3710.30 mW** and **3167.09 mW**, respectively.

7.1.4 Comparative Summary. GPT-2 outperformed the other models in all assessed metrics, demonstrating optimal balance between performance and resource usage.

Although Open_Llama and Sheared_Llama are more complex models, their increased resource requirements did not translate into proportionate performance gains in this testing scenario.

7.2 Discussion

The surprising efficiency of **GPT-2** suggests that optimization and architectural considerations can significantly influence model performance, potentially more than model complexity alone. This finding is critical for applications where both performance efficiency and resource management are key considerations, particularly in edge computing environments.

Open_llama and **Sheared_llama** might require further optimizations or may be more suitable for environments with less stringent resource constraints where the focus can afford to shift towards achieving higher accuracy.

These insights underscore the importance of selecting the right model based on the specific operational environment and application requirements, as well as the practical implications of model deployment in real-world scenarios.

8 LINK TO SOURCE AND OTHER RESOURCES [3 PTS]

This section provides links to the public repositories, scripts, and tutorials that were utilized in the development and benchmarking of the project. These resources are essential for replicating the research and extending the findings presented.

8.1 Project Files

8.2 Model Resources

(1) Sheared-LLaMa-2.7B: https://huggingface.co/princeton-nlp/Sheared-LLaMA-2.7B
(2) OpenLLaMa 3B v2: https://huggingface.co/openlm-research/open_llama_3b_v2
(3) GPT-2: https://huggingface.co/openai-community/gpt2

8.3 Tutorials and Setup Guides

(1) NVIDIA Tegrastats Utility Guide: https://docs.nvidia.com/jetson/archives/r35.4.1/DeveloperGuide/text/AT/JetsonLinuxDevelopmentTools/TegrastatsUtility.html
(2) Jetson AI Lab Model Setup: https://www.jetson-ai-lab.com/tutorial_slm.html

8.4 Benchmarking Tools

(1) Jetson Containers for LLMs: https://github.com/dusty-nv/jetson-containers/tree/master/packages/llm/transformers

8.5 Project Github Repository

(1) LLM Inference on Edge: https://github.com/jolinesikora/LLM_Inference_on_Edge/tree/main
(2) Run Tests Batch file: https://github.com/jolinesikora/LLM_Inference_on_Edge/blob/main/run_tests.bash
(3) Process Tegrastats file to summarize power consumption https://github.com/jolinesikora/LLM_Inference_on_Edge/blob/main/Calculate_Tegrastats.py

9 PROJECT/CLASS TAKEAWAY [2 PTS]

This project has enriched our understanding of performance analysis and computing acceleration. One key takeaway was mastering hardware-specific optimizations, such as utilizing NVIDIA's CUDA and TensorRT technologies. These tools are essential for enhancing operational efficiency, reducing computational overhead, and improving response times across various applications.

The hands-on experience with real-time performance monitoring, using tools like 'tegrastats', has given us a practical grasp of resource utilization and its impact on system performance. This is great for any high-resource application and will be invaluable in our future careers should we need to know about resource utilization. As we look forward to roles involving the deployment of complex computational solutions in real-world scenarios, the ability to optimize system performance to ensure efficient operation under diverse conditions will be helpful.

Lastly, this project highlighted the importance of integrating both theoretical knowledge and practical skills in computing acceleration. This balanced perspective will be beneficial as we aim to bridge the gap between development and deployment in our professional endeavors.

10 FEEDBACK [+1 PTS (GOES TO PARTICIPATION)]

The biggest thing that would have improved this class would have been to have more time to work on our projects. Having access to the list of devices, and being able to pick our projects within the first 3-4 weeks of class, and gaining access to the hardware by week 5, and giving our presentations by week 6, and starting our research by week 7 or 8 would have made us feel a lot more comfortable and less stressed about the timing. Having everything accelerated at the end of class, with the learning curve and research required, while also working on final projects and preparing for finals in other classes, was overwhelming.

We did appreciate your teaching style and how approachable you are. Thank you for everything.

REFERENCES

[1] Xinyang Geng and Hao Liu. 2023. *OpenLLaMA: An Open Reproduction of LLaMA.* https://github.com/openlm-research/open_llama
[2] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149* (2015).
[3] Jangho Kim, Simyung Chang, and Nojun Kwak. 2021. PQK: Model Compression via Pruning, Quantization, and Knowledge Distillation. In *Proc. Interspeech 2021.* 4568–4572. https://doi.org/10.21437/Interspeech.2021-248
[4] Yi Ma et al. 2018. Sparse Learning for Large-scale and High-dimensional Data: Algorithms, Theory, and Applications. *arXiv preprint arXiv:1804.04010* (2018).
[5] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019).
[6] Yiding Wang, Kai Chen, Haisheng Tan, and Kun Guo. 2023. Tabi: An Efficient Multi-Level Inference System for Large Language Models. *EuroSys Conference* (2023).
[7] Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2023. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694* (2023).