Scalably Serving LLMs for Research and Education

In recent years, Large Language Models (LLMs) have become an essential component in various domains, including research, education, and industry. These models have the ability to understand and generate human-like text, making them a valuable tool for natural language processing tasks. This paper presents a system designed and implemented to serve LLMs to many users efficiently, leveraging parallel computing and datacenter techniques. The system was put through its paces by addressing the needs of 2 graduate researchers, who wanted to use LLMs to categorize accounting data descriptions effectively and at scale.

Ptolemy, a cluster constructed for the purpose consists of 8x Nvidia DGX-like 8xA100 nodes in a standard Slurm cluster. The Slurm scheduler is used to orchestrate the number of backends, ensuring efficient resource utilization. The primary components of the system are the backend, the load balancer, and the frontend. The backend, llama.cpp, is used as the implementation for the LLM, employing various quantized models for flexibility and efficiency. Haproxy, a tried-and-true load balancer, is responsible for distributing incoming requests across multiple Llama.cpp instances, the key in handling simultaneous inferencing requests. Special care was taken to configure Haproxy to suit the usage tendencies of llama.cpp. OpenWebUI, a user-friendly web interface, provides an attractive frontend for users, offering chats, web search, rag, and coding functionalities, among others. It also ties into multiple backends, including image generation backends such as ComfyUI. Communication between components is managed through REST/HTTP requests, allowing for flexibility in load balancing at all stages of the pipeline.

At the onset, it was proposed to dedicate a node from Ptolemy for this purpose of serving LLMs. However, after testing, it was determined that maintaining all nodes in the cluster and utilizing the Slurm scheduler yielded better results for our specific use-case. The primary challenge I encountered was load balancer instability under high concurrency when using Paddler. In Initial attempts when using Paddler as the load balancer, loads with high concurrency would result in panics where all requests were dropped. To address this, Haproxy was implemented, which proved to be more reliable and efficient. The system was configured, optimized, and tested to ensure it could handle the researchers' needs, including datasets of 10k to 500k.

A few strategies could be implemented to further performance and usability. Such strategies include further horizontal scaling with more GPUs or nodes, and prompt

caching. Future enhancements could involve integrating more advanced load balancing algorithms or better resource management strategies.

This paper presents an overview of a scalable system for serving LLMs to many users efficiently. A proof-of-concept system was successfully deployed and has proven to be a valuable resource for the researchers collaborating on the project. The broader implications of this work suggest that the system could serve as a valuable resource for other researchers and students working with LLMs.