Zhiyu Zhao

Education

Nanjing University Sep. 2023 - Jun. 2026

Atmospheric Science Master of Science

Nanjing University Sep. 2019 - Jun. 2023

Atmospheric Science Bachelor of Science
GPA: 4.66/5.00 (Rank 2/65)

Technical Skills

- Proficient in CUDA programming, understanding optimization techniques for common LLM operators and basic GPU hardware architecture.
- Proficient in C++ programming, familiar with C++11 features, and experienced with build tools like CMake and Bazel.
- Proficient in Python programming, skilled in PyTorch deep learning framework, and familiar with vLLM inference framework.
- Familiar with common LLM inference acceleration algorithms such as PagedAttention, FlashAttention, and KV Cache.
- Fluent in English, capable of reading English literature and technical documentation; CET-6 score 602.

Internship Experience

Sensetime | HPC Department | Systems Research Intern May 2024 - Nov. 2024

Responsibilities Involved in the development of PPL Serving and OPMX, responsible for the development and performance testing of LLM inference systems.

- Optimized PPL serving scheduling policy module, improving QPS by 10% (C++).
- Developed OPMX, a PPL weight conversion tool (Python).
- Designed and developed the inference performance testing system **llm.perf**, simulating performance differences of various inference engines under high-concurrency and long-context loads.
- Conducted large-scale performance tests based on llm.perf, covering models like LLaMA 2/3, Qwen2, Deepseek V2; GPUs including L40S, A100, A800, H800; and inference engines such as vLLM, SGLang, LightLLM, and PPL.

NVIDIA | Autonomous Vehicle Platform | Software Intern Dec. 2024 - May 2025

Responsibilities NVIDIA DriveWorks software platform development. Responsible for setting up code static analysis and log whitelisting infrastructure, and for Coverity and Test coverage of image processing modules.

- Built a code static analysis system based on Python and Bazel to control code quality and integrate with the CI/CD system.
- Developed a log whitelisting system based on Python, enabling automatic testing, synchronization, and tracking of whitelists, and integrating with bug tracking tools.
- Wrote CUDA test cases to improve the test coverage of the image processing module.
- Fixed Coverity violations in the codebase, ensuring compliance and security of C++ code.

Alibaba Cloud | PAI | Systems Research Intern May 2025 - Present

Project Experience

Inference Framework Performance Testing System - llm.perf

Project Description Designed and implemented a benchmark system for testing LLM inference framework performance in real-world scenarios. Supports static and dynamic performance testing of various LLM inference frameworks, including evaluation of key metrics such as throughput and latency.

- Supports performance evaluation and comparison of multiple inference backends (vLLM, LightLLM, SGLang, PPL, etc.).
- Static inference tests evaluate native engine performance, while dynamic tests assess overall online service performance.

- Supports two testing scenarios: fixed concurrent users and fixed request rate, simulating large-scale concurrent requests via multi-threading/multi-processing.
- Supports a comprehensive performance metric evaluation system, including QPS (Queries Per Second), TTFT (Time to First Token), TPOT (Time Per Output Token), E2E (End-to-End Latency), ITL (Inter-Token Latency), TPS (Throughput), etc.
- Customizable parameters via configuration file, including batch size, model, TP, PP, EP, input/output length, concurrency, warm-up time, etc.
- Designed unified test datasets and evaluation methods to ensure fair and effective performance comparison between different backends.

YOLOv8-based Industrial Meter Visual Reading System

Project Description Designed and implemented an industrial meter reading system capable of automatically extracting readings from liquid level gauges and pointer-type barometers from RTSP video streams. Achieved efficient real-time processing on an NVIDIA Tesla P4 GPU through TensorRT acceleration and CUDA optimization.
Project Link https://github.com/lantel-wm/meter_infer

- Trained YOLOv8n (3.2M) and YOLOv8s-seg (11.8M) models using thousands of real industrial meter images as a dataset.
- Developed a dual-model pipeline: input images successively underwent pre-processing, object detection, semantic segmentation, and post-processing to extract meter information.
- Designed image processing algorithms to obtain dial scale and pointer positions from detection and segmentation results, thereby deriving meter readings.
- Implemented a C++ multi-threaded producer-consumer processing framework supporting multi-channel concurrency, capable of handling multiple RTSP video stream inputs.
- Utilized CUDA for pre-processing acceleration and TensorRT for model inference optimization, achieving an overall frame rate of 20 FPS when concurrently processing 8 video streams.

Lightweight LLM Inference Framework

Project Description Designed and implemented a lightweight large language model inference framework supporting both CPU and CUDA GPU backends. Provided efficient model inference experience through CUDA kernel optimization and fine-grained memory management.
Project Link https://github.com/lantel-wm/llm_infer

- Implemented complete Transformer architecture components, including MHA, RoPE, FFN, etc.
- Developed high-performance CUDA operators, including GeMM, Rotary Position Embedding, RMSNorm, etc.
- Implemented KV-Cache mechanism to accelerate sequence processing during auto-regressive generation.
- Implemented the core framework using C++20 and CUDA, optimizing CPU backend performance with the Armadillo linear algebra library.
- Adopted a modular design, providing a unified Tensor abstraction and hierarchical operation interface, supporting flexible expansion of model structures.

CMU 10-714 Deep Learning Systems

Project Description Developed a simplified PyTorch-like deep learning framework from scratch (without using NumPy) using Python, C++, and CUDA.

- Implemented dynamic computation graph generation and forward computation, and an automatic differentiation system based on topological sorting.
- Wrote PyTorch-like neural network modules, such as nn.Linear(), nn.LayerNorm1d(), nn.Dropout(), etc.
- Implemented random initialization of neural network weights, and wrote SGD and Adam optimizers.
- Developed an NDArray library in C++ and CUDA as the backend for Tensors, implementing operations like reduction and matrix multiplication using CUDA.
- Implemented the training of common deep learning models such as CNN and Transformer using this simplified deep learning framework.