Ziyun CHENG

(551) 267-0191 | ziyun_cheng@outlook.com | linkedin.com/in/zy-cheng/ | searching for 2025/2026 fulltime

Education

New York University M.S. CS GPA: 3.9 New York, US 2023.9 - 2025.12 (Expected)

- **Selected Courses:** ML System, Cloud and ML, Devops, LLM & Vision Model, AI and Hardware Accelerator, etc.

Shanghai Jiao Tong University B.S. CS General GPA: 3.7 Shanghai, China 2019.9 - 2023.7

Skills

- **Languages**: C/C++, Python, Golang, SQL, Java, JavaScript, Scala, HTML/CSS, Shell
- **Frameworks & Tools**: PyTorch, JAX, CUDA, Compiler (XLA, Triton, MLIR, LLVM), Parallel Communication (MPI, NVSHMEM, NCCL), Database, Docker, Kubernetes, Jenkins, Linux, GCP, AWS, LLM.

Internship

Nvidia, XLA | *DL Compiler Intern | C++, JAX, HLO, MLIR, NVSHMEM, TP Inference* 2025.6 - 2025.11 (Expected)

- Implement new feature: Compute - **Distributed Communication** fusion within kernel on multi-GPU parallel condition via **NVSHMEM**, minimize launch and data movement overhead, take TP inference all-reduce + softmax case as example [PR].
- E2E compile time Implementation: Registered custom **HLO** passes to fuse operations in computation graph; introduced custom AllReduce placeholders, extended **Triton** TTIR graph generation logic; implemented actual NVSHMEM device API calls within MLIR framework; and enable automated linkage of the NVSHMEM library in the **PTX** backend.
- Added memory coloring and assignment logic to support NVSHMEM symmetric memory initialization and runtime.

Nvidia, Compute Arch | *Compute Arch Intern, CuTile Team | C++, System, MLIR, Triton* 2024.7 - 2025.1

- Developed **Triton** front-end compiler on **Blackwell** Architecture (B100), switch triton ttgir backend to nv tileIR backend; designed tileIR convert pass (C++) from triton TTIR to lower layers such as **CuTile** with **MLIR** framework.
- Added Triton extended syntax, enabled python level programmable warp specialization (pws) between load/store and compute to accelerate, rewrote high-level test case such as Batch Norm and **Flash Attention**, outperform OpenAI.
- Supported new IR Opeartors adding, fix compile and runtime errors such as address misalign, load/store vectorization, etc.
- Responsible for infra part, built **Jenkins**-based CIs to avoid regression, enabled auto-run functional / performance tests with multicore parallelism; lock equipment frequency to calculate performance, conduct analysis with **Ncu**/**Nsys**.

ByteDance | *Machine Learning Engineer Intern | Pytorch, Python, Infra* 2023.2 - 2023.7

Applied Machine Learning Team | Scientific Computing, AI Platform (now reorg into Seed)

- Trained models for QSAR with multi-dimension molecule data, utilized techniques such as **GCN** and **Transformer**.
- Used parallel file system vePFS for data storage (Tbs), defined auto-train workflow with **YAML**, performed **distributed data parallelism** with **PyTorchDDP** due to large dataset, set up infrastructure for efficient computing.
- Designed **MySQL**-based data platform for inner use, realized 2-levels authority, input data with PDF/Excel format.

Research Experience

New York University, NY | *Student Research Assistant, Open Source Contributor | NYU System Lab* 2024.1 - 2024.6

Grendel-GS [github] | *Pytorch, C++, CUDA, Distributed Training | Advisor: PhD Hexu Zhao & Prof. Jinyang Li*

- Introduced Grendel, a distributed trainging system designed to partition 3D Gaussian Splattinga (3DGS, 3D reconstruction method available on single GPU) to multiple GPUs; supported for multi-batch size, enabled **data and tensor parallelism**.
- Realized **dynamic load balance** of parameters through all to all communication, automatically adjusted training hyperparameters based on batch size, achieved 27.28 PSNR for 40.4 million Gaussian functions on 16 GPUs.
- Devoted in loss calculation, fused SSIM with L1 loss into one Pytorch Operator with **CUDA**, reduced time to 54.2%.

Selected Projects

LayerSkip-Sparse: Sparsity based Self-Speculate Decoding [github] | *LLM Inference, Pytorch* 2025.2 - 2025.6

- Enabled E2E sparse draft model inference: introduced “enable_sparsity” and “sparsity_algo” options to combine structured sparsity (e.g. 4:2) with adaptive self-speculative decoding, also support loaded sparsity masks to generate draft model.
- Redesigned KV cache strategy for sparse attention scenarios by adding a configurable option to disable KV cache reuse.

LLMChat: LLM Inference based Chat Robot [github] | *TRT-LLM, Inference, Deployment* 2024.1 - 2024.5

- Developed a chat robot based on **TensorRT-LLM, Triton inference service,** and **React;** tested **TP, PP** and quantisation results on different combination of GPUs, optimized with **prompt** engineering, enable **multi-turn conversations.**
- Packaged with **Docker**, deployed on **Kubernetes** cluster; supported autoscaling with **Helm**, enable Slack notification.

CUDA Version Handwritten Digit Recognition MLP [github] | *C++, ML, CUDA* 2024.1 - 2024.5

- Implemented a handwritten recognition MLP model from scratch using **C++** and **CUDA**, including tensor definition, CUDA operation for tensor (elementwise, tiled matmul, reduction), fused operator to calculate cross entropy loss.
- Constructed a C++ MLP class, including multi-layers, forward / backward; instead of autograd, performed a manual gradient calculation to get a better performance; applied to the MNIST dataset, achieved an accuracy of 98.26%