Jierui (Jerry) Xu

Tel: 608-395-1979 | Email: xjr2423@gmail.com | LinkedIn: Jierui Xu | GitHub: Jerry2423

Education

University of Wisconsin–Madison Madison, WI

Bachelor of Science in Computer Science — GPA: 4.0 / 4.0 *Sep. 2024 – May. 2027*

- Coursework: High Performance Computing (A, graduate-level), Operating Systems (ongoing)

ShanghaiTech University Shanghai, China

Bachelor of Engineering in Computer Science — GPA: 3.9 / 4.0 *Sep. 2022 – May. 2024*

- Coursework: Intro to AI (A+, graduate-level), Computer Architecture (A+)

Interests & Skills

- **Interests**: My research interest lies in building high-performance and scalable systems for LLMs. I focus on software-hardware co-design for novel AI accelerator architectures (like AWS Trainium and GPUs) to minimize latency and maximize throughput at scale.
- **Languages**: Python, C++, C, Java, JavaScript/TypeScript, SQL, HTML, CSS, LaTeX
- **Models, Tools & Frameworks**: PyTorch, TensorFlow, Hugging Face, Transformers, CUDA, Triton, TensorRT, vLLM, ONNX

Experience

University of California Merced Jan. 2025 – Oct. 2025

Research Assistant (Advisor: Prof. Dong Li) in Collaboration with Amazon Web Services (AWS) *Remote*

- Optimized Llama 3.2 1B inference with custom kernels on AWS's AI accelerator (Trainium), achieving 78% latency reduction (6.43s → 1.40s) and 4.8x throughput (102.60 → 494.39 tokens/s) vs. PyTorch baseline.
- Compressed Llama and Qwen LLMs via Singular Value Decomposition (SVD), then applied LoRA fine-tuning to restore performance, limiting the mean accuracy (mAcc) drop to ≤ 0.10 across 9 datasets.
- Redesigned fused attention kernel with tiling techniques to optimize tensor layouts on SBUF and PSUM memory, expanding maximum sequence length capacity by 7.8× (from 640 to 5k tokens).

Amazon May. 2025 – Aug. 2025

Software Engineer Intern *Shanghai, China*

- Designed and implemented an LLM-based search keywords recommendation system that analyzes real-time customer behavior to generate search suggestions, driving $7.12MM annualized operating profit.
- Automated an LLM inference platform on AWS ECS clusters with Triton server and vLLM backend, achieving sub-100ms latency at scale.
- Developed a daily automated Spark SQL pipeline processing 5M+ customer clickstream events to analyze shopping patterns, saving 4+ hours of manual data engineering maintenance per week.

Publication

NeuronMM: High-Performance Matrix Multiplication for LLM Inference on AWS Trainium

Dinghong Song, **Jierui Xu***, Weichu Yang, Pengfei Su, Dong Li*

- Submitted to European Conference on Computer Systems (EuroSys) 2026 [code]

** indicates equal contribution*

Awards

ASPLOS / EuroSys 2025 Programming Contest Apr. 2025

Second Place Winner *Rotterdam, The Netherlands*

- Awarded for developing the fastest inference implementation of the Llama 3.2 1B model on AWS Trainium hardware by designing highly-optimized custom kernels using the Neuron Kernel Interface (NKI).

ICPC 2024 North Central North America Regional Contest Nov. 2024

*Ranked in the **Top 10**, out of 250+ contestants.* *Madison, WI*

- Collaborated with teammates and solved algorithm and data structure problems in real-time under tight time constraints.

Research Work Reproduction

Language Models are Unsupervised Multitask Learners - OpenAI: GPT-2

- Leveraged Distributed Data Parallel (DDP) training across 8x A100 GPUs to reproduce OpenAI's GPT-2 (124M).
- Trained the model on the FineWeb dataset, achieving the 2.9 validation loss.

FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning - ICLR 2024

- Developed custom forward and backward pass attention kernels in Triton to implement the FlashAttention-2.
- Optimized GPU work partitioning and leveraged mixed-precision (FP16/FP8) to minimize memory I/O operations and maximize on-chip parallelism.

Projects

Shakespeare GPT | *PyTorch, HuggingFace Datasets*

- Engineered a complete GPT framework from scratch in PyTorch, implementing a modern Transformer architecture (w/ RoPE, RMSNorm), a BPE tokenizer, and an AdamW optimizer.
- Trained the model on a Shakespearean corpus (1.5 hrs, V100) to generate Shakespeare-style text, achieving a 3.55 validation loss.

CUDA LLM Inference Engine | *C++17, CUDA, Jenkins*

- Developed a CUDA-based command-line application to run LLM, generating texts with user-defined prompts.
- Developed self-attention kernel, and used cuBLAS for optimized matrix operations, boosting performance from 110 tokens per second to 520 tokens per second on NVIDIA RTX 4050.
- Built an automated test suite with memory error detection support based on Valgrind and Jenkins, discovering and fixing 18 bugs.

Selected Readings

- Wang, X. et al. (2025).*SVD-LLM: Truncation-aware singular value decomposition for large language model compression*.
- Dao, T. (2024). *FlashAttention-2: Faster attention with better parallelism and work partitioning*.
- Kwon, W. et al. (2023). *Efficient memory management for large language model serving with pagedattention*.
- Hu, E. J. et al. (2022). *LoRA: Low-rank adaptation of large language models*.
- Rajbhandari, S. et al. (2020). *ZeRO: Memory optimizations toward training trillion parameter models*.
- Shoeybi, M. et al. (2019).*Megatron-LM: Training multi-billion parameter language models using model parallelism*.
- Vaswani, A. et al. (2017).*Attention is all you need*.