Peterson Guo

peterson.guo@uwaterloo.ca | petersonguo.com | PetersonGuo | PetersonGuo

Education

University of Waterloo — Waterloo, ON
BASc. in Honours Electrical Engineering — *09/2023 - 04/2028*

Skills

Languages: C/C++, Python, MLIR, JS, Bash, Java, SQL, Terraform
Software: Git, LLVM, PyTorch, CUDA, ROCm, NumPy, Pandas, Docker, AWS
Others: Compilers, Drivers, Operating Systems, Linux, LLMs, CNNs, LSTMs, Neural Networks, Kernel Debugging

Experience

Systems Software Engineer Intern — 05/2025 – 12/2025
Nvidia — *Toronto, ON · Santa Clara, CA*

- Reduced **JIT** compilation time by ∼**19.5%** by designing a deterministic **parallel** Merkle-tree hashing system ∼**5×** faster than SHA256 for multi-GB binaries, improving kernel generation **latency** and iteration speed
- Cut **kernel launch** overhead by ∼**60%** by profiling the CUTLASS compiler/runtime pipeline, removing slow paths, and applying targeted **micro-optimizations**, significantly reducing **Python→C++** dispatch cost in launch-bound workloads to within **10%** of competitors
- Built core infrastructure for Nvidia's **MLIR**-based **CUDA** dialect, adding **20+** IR ops, enums, and lowering patterns (CUDA Graphs and streams, async memops, pointer ops) enabling **Python** DSL users to target advanced features on **Rubin, Feynman, and Blackwell GPUs** ahead of upstream **MLIR**
- Improved **GPU throughput** in **LLM** workloads by converting Blackwell-tuned kernels that reduce launch overhead and improve integration with JIT kernel generation
- Performed end-to-end **latency** decomposition using Nsight Systems, Nsight Compute, CPU profilers, and Perfetto, isolating launch cost, Python overhead, compiler time, **contention**, scheduling stalls, and memory-stall contributors to guide fixes across the stack
- Delivered up to **2.67×** faster **LLM inference** by integrating custom kernels into a **Python**-based **ML JIT**-compiler, reducing memory traffic and launch volume, and improving fusion heuristics for an additional **0.7%** speedup on **multi-billion** parameter models

Software Engineer Intern — 09/2024 – 12/2024
AMD — *Markham, ON*

- Shaped display-pipeline research, shown by my **ML upscaling** PoC being adopted and later expanded into an internal technical conference paper by my successor, by building a lightweight **PyTorch/ROCm** prototype
- Strengthened display-pipeline stability on next-gen AMD GPUs/APUs by building **C/C++ kernel** drivers, ensuring robust framebuffer→display behavior for Ryzen AI and Radeon hardware
- Delivered AMD's most stable GPU software release, by resolving **25+** kernel-level crashes, hangs, and performance regressions, using WinDbg, crash dumps, ETL traces, **firmware** logs, and **register-level** analysis
- Improved bring-up reliability by fixing **5+** Microsoft OS–GPU issues, analyzing **firmware**/memory dumps and eliminating early-boot bottlenecks
- Enhanced **Linux** display pipeline correctness, by resolving defects in color calibration, frame-sync logic, DSC decoding, and corruption, contributing patches to AMD's **open-source** Linux display driver

Security Developer Co-op — 01/2024 – 5/2024
eSentire — *Waterloo, ON*

- Cut **ingestion latency** by **50%+** by **optimizing** JSON parsing and Snowflake execution paths, reducing pipeline stalls in **real-time analytics**
- Improved log-processing throughput by **4×** through **algorithmic optimizations** in **Python**'s data pipeline
- Built an AI threat analytics dashboard end-to-end (Snowflake, Python, Vue3) used by enterprise clients to investigate live security events, earning executive visibility
- Strengthened analyst workflows by adding packet-processing and correctness enhancements to an open-source PCAP scrubber

Projects

ML Upscaling | ROCm, CUDA, Machine Learning, CNNs, PyTorch, Python

- Prototyped a real-time **ML** upscaling model for **AMD graphics cards**, leveraging **transformer** based **super sampling** to enhance visual fidelity and performance for potential display pipeline adoption

InvestIQ | Python, LSTMs, IBKR API, CUDA

- Built **CUDA**-accelerated **LSTM** trading system with real-time market data using IBKR API, incorporating **Monte Carlo** volatility forecasting

Bionic Evo | C/C++, Assembly, Neural Networks, TensorFlow, CUDA, STM32

- Engineered a humanoid arm prototype for amputees by utilizing **STM32** and **EMG** sensors, integrating **pattern recognition** to achieve **precise gesture classification** and seamless arm control