Minseo Choi

410-301-6057 | cfi3288@gmail.com | linkedin.com/in/minseoc03 | github.com/minseoc03 | minseoc03.github.io

Education

Johns Hopkins University Baltimore, MD
Bachelor of Science in Computer Science *Expected May 2028*
Relevant Coursework: Parallel Programming, Performance Engineering, Compilers, Machine Learning, Deep Learning, Operating Systems, Databases, Algorithms, Data Structures
Self-Study: GPU Kernel Optimization, Inference Optimization, ML Compiler Design, LLM Serving, Distributed System, Computer Vision, Natural Language Processing, OS Resource Management, Profiling & Debugging

Technical Skills

Languages: C, C++, Python, Java, SQL, x86-64 Assembly
ML & Inference: PyTorch, HuggingFace, Triton-Lang, CUDA, vLLM, SGLang, TensorRT-LLM, Triton Inference Server
Compilers & Build: MLIR, LLVM, TableGen, CMake, Bazel
Infra & Tools: Prometheus, Grafana, SLURM, Supabase, Vercel, Git, Linux, Shell

Experience

Johns Hopkins Data Science and AI Institute (DSAI) May 2026 – Present
Research Assistant – BLAB *Baltimore, MD*
- Profiled decode-bound rewriting workloads in a large-scale data generation pipeline and identified high copy-fraction as an opportunity for draft-free speculation
- Deployed suffix decoding, achieving 1.43× throughput improvement without additional model training or VRAM overhead

Johns Hopkins Medicine Feb. 2026 – Present
Research Assistant – PALS (Psychiatric-Assistive Learning Systems) Lab *Baltimore, MD*
- Built a multi-agent LLM safety-labeling pipeline serving six models (a 70B conversation generator and five role-aligned critic agents (2×34B, 3×8B); all FP8 quantized) across 2×B200 GPUs using TensorRT-LLM and Triton Inference Server
- Diagnosed orchestration-bound GPU underutilization and improved decode throughput 2.89× via generator GPU isolation, async BLS execution enabling TensorRT-LLM in-flight batching, and KV-cache block reuse over shared Monte-Carlo prefixes
- Achieved ∼3.3 s P95 end-to-end latency for auto-labeled cases and ∼13 s P95 for cases escalated through specialist agents (Triage → Legal/Risk), processing ∼3,100 conversations/hour (∼86% auto-labeled, ∼14% terminating in clinician review)
- Extending to heterogeneous engine serving (SGLang + TensorRT-LLM): routing prefix-heavy safety-policy prompts to SGLang's RadixAttention, with ongoing research on cross-engine KV cache transfer
- In collaboration with NVIDIA Safety and Johns Hopkins Technology Ventures (JHTV); ICML 2027 submission planned

Johns Hopkins University Jan. 2026 – Present
Course Assistant – Computer Systems Fundamentals *Baltimore, MD*
- Course Assistant for EN.601.229 Computer Systems Fundamentals — mentoring students and debugging code on cache behavior, memory hierarchy, assembly, concurrency, and performance bottlenecks in C/C++ on Linux

Republic of Korea Army Oct. 2023 – Apr. 2025
Drill Instructor & Senior Squad Leader *Nonsan, Republic of Korea*
- Trained and mentored 2,000+ recruits as Senior Squad Leader, coordinating large-scale operations under high-pressure
- Developed resilience, rapid decision-making, and accountability — transferable to ambiguous research environments

Project

Personal Technical Blog (minseoc03.github.io) Jan. 2024 – Present
- Author of a technical blog on ML systems, GPU runtimes, and AI compiler infrastructure
- Explore hardware–software co-design trends in emerging AI accelerators
- Write technical essays translating systems-level research into practical implementation insights
- **Notable Posts** : Deep Dive on FlashAttention 4, Future of Low-Latency AI Inference

ML Systems & **Compiler Stack** Aug. 2025 – Present
- Built FlashAttention 2 from scratch in Triton/CUDA with tiled online-softmax kernels, eliminating $O(N^2)$ materialization for $\sim 2\times$ speedup and $\sim 5\times$ peak-memory reduction over naive PyTorch attention ($seq\_len = 4K$)
- Built a toy LLVM compiler and extended it with IR/CFG visualization and optimization passes
- Developed an end-to-end MLIR pipeline including IR parsing, pass execution, and toolchain integration
- **GitHub** : github.com/minseoc03/mlsys-self-study

Medical Image Enhancement Acceleration Oct. 2025 – Present
- Built GPU-accelerated CT/MRI enhancement pipeline combining DICOM preprocessing, DL denoising, and post-processing
- Developed fused Triton kernels for DICOM pre/post-processing chains (dtype conversion, windowing, normalization, clipping), achieving ∼9.7× speedup over unfused PyTorch ops by eliminating per-op kernel-launch/dispatch overhead and intermediate HBM round-trips
- **GitHub** : github.com/minseoc03/medical-enhancement-triton