Ronit Anilkumar

(707) 236-4722 | ranilkumar@ucla.edu | linkedin.com/in/ronit-anilkumar | ronitanilkumar.com

Education

University of California, Los Angeles (UCLA) Los Angeles, CA
Bachelor of Science in Computer Science and Engineering *Expected June 2027*

Coursework

- Computer Systems Architecture
- Operating Systems
- Algorithms & Complexity
- Distributed Systems
- Computer Network Fundamentals
- Data Structures & Algorithms

Experience

Machine Learning Researcher | UCLA Biomedical AI Research Lab Nov. 2024 – Present
University of California, Los Angeles *Los Angeles, CA*

- Optimized distributed training throughput **1.6x** on A100 GPU clusters via mixed-precision training and GPU profiling; co-authored US-JEPA, a self-supervised foundation model for ultrasound cancer detection trained on **4.73M** frames across **49** datasets.
- Built fault-tolerant distributed data pipelines processing **4.73M+** ultrasound frames across **8+** anatomical domains with automated quality validation, preprocessing workflows, and domain-specific augmentation, enabling reproducible large-scale ML workflows.

Projects

Loom | GitHub | *Rust, Tokio, Axum, Reqwest, SHA-256* May 2026 – Present

- Built a concurrent model weight streaming agent in **Rust** that parallelizes HTTP Range requests across **16 workers** to eliminate sequential latency when fetching models from object storage; on high-latency networks (S3, GCS), concurrent requests reduce 270 serial round-trips (40s of blocking wait) to pipelined latency (**2-3s**).
- Implemented a content-addressable disk cache indexed by **SHA-256** hashes with hard-link assembly; cache hits resolve in **41 ms** (file-accessible time via hard-link), avoiding redundant storage and enabling fast model re-initialization across containers.
- Designed a lightweight **Axum** control plane that tracks per-node cache state, enabling cold-start containers to query peer caches before falling back to remote storage, reducing initialization latency in heterogeneous inference clusters.

Veylor | GitHub | *LangGraph, Python, FastAPI, Docker, OpenTelemetry, Semgrep* June 2025 – Present

- Designed and built an autonomous multi-agent CI/CD repair system using **LangGraph** orchestration with specialized agents for fault localization, patch generation, and test validation, enabling end-to-end automated repair without human intervention via **GitHub Actions** integration.
- Built security-hardened **Docker** sandboxes with strict resource limits (512MB RAM, 1 CPU, 60s timeout), no-network policies, and **Semgrep** CVSS v3.1 static analysis enforcing zero vulnerabilities across **1,000+** patches validated.
- Implemented **OpenTelemetry** distributed tracing to measure system performance, tracking p50/p95/p99 latencies and real-time CPU/memory utilization across the multi-agent pipeline to identify bottlenecks and optimize resource allocation.

CoWrite | GitHub | *Yjs, TipTap, WebSockets, SQLite, TypeScript, Node.js, Anthropic API* March 2026 – Present

- Built a real-time collaborative rich text editor using **CRDT-based** sync (**Yjs**) and **WebSocket** broadcast with conflict-free concurrent edits, live cursor presence, and offline-first reconciliation on reconnect.

Technical Skills

Programming Languages: Python, Rust, C++, TypeScript, JavaScript, SQL, Verilog
AI/ML: PyTorch, distributed training, mixed-precision optimization, GPU profiling, A100 GPU clusters
Systems & Databases: Distributed systems, CRDTs, WebSockets, PostgreSQL, Redis, SQLite, performance observability
Cloud & Infrastructure: AWS, Docker, Linux, GitHub Actions, CI/CD, OpenTelemetry, Kubernetes