Kevin A. Li

kevin.li.actual@gmail.com | (281) 728-8334 | https://www.linkedin.com/in/krazykevinli/ | https://github.com/kevinli22527

EDUCATION

The University of Texas at Austin (Electrical and Computer Engineering) **GPA: 3.91/4.0** **Dec 2022**

Relevant Courses: Software Design, Software Testing, Algorithms, Computer Architecture, Computer Vision, Cybersecurity

EXPERIENCE

Performance Software Engineer at Arm **Spring 2023 - Current**

- Built our department's workload-similarity analysis platform, utilizing PCA, cosine similarity, and other proprietary methodologies to classify workloads (both internal and external) and eliminate redundant workload tuning efforts
- Designed our team's Ansible-based benchmark automation tool, which now supports 25+ version-controlled, CI-tested, and result-templated workloads such as MLPerf, Whisper, llama.cpp, SPEC ® 2017, x264/5, HPCG, and loaded-latency
- Designed a genetic algorithm-based automated tuning framework which enables benchmarks to "run themselves", mutating next-run control register and software settings until a target score has been achieved or time runs out
- Designed workload profiling system capable of running multiple simultaneous profilers/tracers, including perf, Turbostat, bpftrace, and power. System also executes benchmark "strategies", such as exponential search on a target latency
- Implemented an end-to-end booking system for hardware access, leveraging Python, React.js, and MongoDB to provision access for both human and auto-users. Resource monitoring algorithms periodically recycle unused resources for utilization
- Configured hundreds of servers, switches, and other devices in our test lab, involving soldering, multimeters, 3D printing, UART/RS232 protocols, firmware hacking, and network hard-wiring at times
- Wrote and upkept hundreds of doc-pages and learning resources (e.g. how-to-tool videos), enabling our team of four to support two new senior engineers and five interns onboarding simultaneously with our tooling and perf-methodology

Software Engineering Intern at bp **Summer 2022**

- Worked in a team of four interns to build a full-stack (React, Flask, MongoDB) AWS EC2 web app which projects wind energy savings across global retail sites. Components communicated via custom-built REST API
- AGILE software development through the Kanban/Scrumban process with Azure DevOps
- Worked with a UI/UX engineer to create the React frontend using MaterialUI, Bootstrap, and ReactStrap
- R&D'ed wind and solar computational models, graphing Pandas-sanitized, NumPy-aggregated data on Jupyter notebooks

Controls Engineering Intern at Samsung Austin Semiconductor **Summer 2021**

- Implemented two chemical sensor monitoring Ignition dashboards in Python to track tank levels and reactor composition
- Software is unit-tested and incorporates verbose error handling and logging, ensuring auditability
- Co-hosted weekly stakeholder meetings to receive project direction receive feedback – had to pivot/redesign many times

Product Engineering Intern at Cirrus Logic **Summer 2020**

- Developed Python algorithms and several additional functions to automate aspects of Burn-In testing/translation
- Developed new and debugged existing JMP scripts for use in statistical quality analysis
- Utilized six sigma concepts such as Cp and Cpk to implement Statistical Process Control methodology

ACADEMIC PROJECTS

College Handsign Detector **Spring 2022**

- Webcam-based hand-sign detector able to detect and categorize multiple human hand signs in real-time
- Utilized machine learning (K-Means, Nearest Neighbor, MediaPipe Hands) to perform detection and classification
- Supervised training performed, with NumPy, SciKitLearn, OpenCV utilized for data processing

Online Auction Simulator **Fall 2020**

- Networked and multithreaded Java program simulating an auction server, allowing clients to register and place bids
- Flexible, easily scalable code supporting admin privileges, multiple different consoles, and persistent data storage

ML-Assisted Bots **Summer 2020**

- Created a Python bot for auto-action in an online video game, generating virtual profits with minimal human time cost
- Created a job application bot, drastically cutting down time to apply to boilerplate jobs (thousands of them)
- Bots utilize natural language processing, image processing, open-source machine learning, and mouse/keyboard automation to emulate human actions. The bots utilize media-capture APIs and error logging to document tasks they cannot perform

SKILLS AND ADDITIONAL INFORMATION

documentation, Python, Java, C, C++, Bash, SQL, MongoDB, React.js, Ansible, Terraform, JMP, Arm Assembly, Git, Jenkins, AWS, GCP, Azure, microservices, Docker, Linux kernels, topdown/bottleneck analysis, profiling/tracing, flame graphs, perf counters, control registers, genetic algorithms, interconnect topologies, microbenchmarks, microcontrollers, networking, UI/UX wireframing