Yifan Yang

LinkedIn ·GitHub ·Google Scholar

Email : yyang29@stanford.edu
Mobile : +1-650-788-9903

Education

- **Stanford University** — Stanford, CA
  B.S. in Electrical Engineering — *Sep 2021 – Jun 2024*
  - **Relevant Coursework**: Computer Security, Blockchain, Digital System Design, Machine Learning, Computer Systems, Parallel Computing, Operating Systems, Compiler, Web Dev, VLSI.

Experience

- **Stanford University** — Stanford, CA
  Research Assistant — *Sep 2022 - present*
  - **Hardware Accelerators for Machine Learning**: The project focuses on creating re-configurable hardware to accelerate machine learning. My work focuses on optimizing BERT on hardware prototypes.
  - **Hyperdimensional Computing Paradigm**: The project focuses on finding a computing paradigm for hyperdimensional computing in many applications like ML and unconventional computing.
  - **Hyperdimensional Computing Early Termination**: Optimizing hyperdimentional computation with statistical models, which terminates heavy computation early and guarantee a bounded accuracy drop.
- **Nvidia** — Santa Clara, CA
  DL Infrastructure Engineer Intern — *Jun 2022 - Sep 2022 & Jun 2023 - Sept 2023 & Full-time since Jun 2024*
  - **Kubernetes**: Set up & deployed a Redis HA service. Improved reliability by containerizing and migrating chip verification services from designated server to Kubernetes cluster. Presented new service architecture to the team.
  - **OpenTelemetry Integration**: Set up service monitoring pipeline with OpenTelemetry protocol. Developed API standard for tracing library. Developed an automated monitoring library deployed to production.
  - **Tasker**: An internal-use platform for scheduling and managing job runs for chiplet designs.
  - **Log storage and query at scale**: Built a distributed full log storage engine that is capable of full text search & aggregation & anomaly detection across TBs of data/day for months with millisecond latency. Serving multiple hardware teams.
- **Si-Tech** — Urumqi, China
  Software Engineer Intern — *Jan 2021 - Jun 2021*
  - **AI Work Order System**: Full-stack developed work order AI processing system on China Unicom backend services using Java and Web technologies. The system uses NLP to parse work order requests and compose possible solutions from database. Reduced more than 90% of human work.
  - **Architecture**: Initiated the improvement of the current backend architecture by using Modularization and dynamic module loads. Developed a standard for future modules. Reduced complexity caused by deep coupling.
  - **Workflows**: Developed several command line tools to automate frequently used workflows. Time spent on repeated work reduced by 80%. Designed and implemented load tests for some China Unicom backend services.
- **China Unicom** — Urumqi, China
  Operation and Maintenance Engineer Intern — *Jan 2020 - Mar 2020*

Projects

- **PiAuto**: Open source project that turns iPads into portable car head units with a Raspberry Pi. The project consists of iOS app client and a server that runs on Raspberry Pi which can be powered through the 12V DC on board. The server connects with OBD port of the vehicle, interprets data, and serves using a self-hosted on-board Wi-Fi. The server also includes an AirPlay middleware to support AirPlay audios to the car speakers. **View Project**
- **IoT System for Collecting Vital Signs and Geographic Location Data of Mobile Users**: IoT research project. Designed a system prototype that could help pandemic control. The project was initiated Jan 2020, at the emergence of COVID-19. Proposed the idea of the project and developed the prototype of the system. Paper published on 2020 International Conference on Communications, Information Systems and Computer Engineering(CISCE). Led a 4-person team, assigning tasks to members, managing project progress, scheduling and hosting group meetings, and key role in directing project development.
- **SigNoz**: SigNoz is an open-source application performance monitoring tool. Helped completed some docs on how OpenTelemetry protocol works. **View Project**