Yiran Su

yiransucdr@gmail.com | linkedin.com/in/yiransucdr

6805 Wood Hollow Dr
Austin, TX

Education

- **The University of Texas at Austin** — Austin, TX, USA
  ***M.S.** in Engineering, **Software Engineering and System** major, **GPA**: 3.76/4* — *Aug. 2019 - May. 2021*
- **Sun Yat-sen University, School of Data and Computer Science** — Guangzhou, China
  ***B.E.** in Network Engineering, top 10% in the class.* — *Aug. 2015 - Jun. 2019*
  - **Course Highlights** Data Structure & Algorithms, Operating System, Computer Network, Web Programming

Skills

Programming Language C++, Python, Java, SQL, HTML, CSS, JavaScript, Kotlin, Shell, GoLang
Framework and Tools React Native, Flask, PyTorch, Tensorflow, Docker, MongoDB, Kubernetes

Intern Experience

- **Graduate Research Assistant(C-PAC)** — Dell Medical School, UT-Austin, Jan. 2021 - May. 2021
  - C-PAC is a configurable processing pipeline for functional brain MRI data. My work is focusing on the configurable frontend of C-PAC, using **React.js** (JavaScript) with React-Redux and React-Saga.
- **Coherent Logix Inc.** — Software Dev Intern, May. 2020 - Aug. 2020, Austin, USA
  - Worked on **neural network quantization**, which simplifies the original neural network model, and persists the original model precision.
  - Accomplished **16/8/4-bit** Quantization Aware Training (**QAT**) with **Tensorflow 2** (**Python**).
  - Developed a set of **QAT generation & validation APIs** in **TensorFlow 2** (**Python**) to simplify future applications.
- **Tencent Inc.** — Software Dev Intern, Sept. 2018 - Mar. 2019, Shenzhen, China
  - Focused on a pattern-based natural language parsing framework in **Python** for a task-oriented chatbot.
  - Deployed the parsing framework on a **Tornado** (Python) Server. The framework was able to handle **100k+** user vocal requests per day, with a **27%** latency drop.
  - Created an **automatic** log analyzer in **Python** to evaluate user behavior (customer stickiness, feature performance, etc.)
- **Graduate Teaching Assistant** — EE 422C (**Java**), UT-Austin, Jan. 2020 - May. 2020
  - Led weekly lectures on Java, including polymorphism, Java generic, multithreading, lambda, stream, etc.

Project Experience

- **Share Your Review** — Course project for EE 382V at UT-Austin, Sept 2019 - Dec 2019
  - Worked on a full stack project that requires both frontend (web/Andriod/ReactNative) and backend (response framework /database) implementation. The project goal was to help people to connect with nearby book readers.
  - Designed a **MVC**-styled **Flask** & **Flask-RESTful** (**Python**) backend. **Firebase** and **PyMongo** for database.
  - Accomplished a **web** frontend with **HTML/CSS/Bootstrap** and embedded it into the Flask-RESTful backend.
  - Applied **Kotlin** for the **Android** frontend. Designed a **reusable cardview template** for the team. Utilized **Volley** to handle requests. Kotlin's **Camera & Location APIs** were applied.
  - Implemented a **React Native** (JavaScript) frontend, with Camera, location, sharing and notification features.
- **Sun Yat-sen University Club Information Platform** — Mar 2018 - Jun 2018
  - Designed the app to spread club event information across campus. Contributed as a frontend developer.
  - Constructed the **Android** frontend in **Java**, with **Retrofit2** & **RxJava** applied. Designed the Web frontend in **HTML/CSS/JavaScript**.
- **2018 International Aerial Robotics Competition** — Computer Vision, Sept 2017 - Aug 2018
  - For ground object detection, designed an **OpenCV**-based Support Vector Machine (SVM) classifier, with a self-implemented Histogram of Oriented Gradient (HOG) descriptor in **C++**.

Publications

Manor, L., **Su, Y.**, et al. "**How to apply for financial aid: Exploring perplexity and jargon in texts for non-expert audiences**", SCiL 2021, accepted.

- Proved that pre-trained neural language models are less likely to predict jargon phases without fine-tuning, by analyzing the prediction perplexity of **GPT-2**.
- Examined the computational linguistic association between jargons and the corresponding context with **PyTorch** (**Python**). BERT embeddings were applied. (This phase was not presented in the paper)