Final Report – Suning Yao

Background

I am working as a Software Engineer intern at the CoreML Device API team at Google Cloud. The team is the owner of ML Infrastructure Runtime & API.

My intern mode is hybrid.

Project

My project is called “Peer-assisted HashMap”. The task is to build a hashmap library, which can be used to share immutable memory objects easily among systems (peers) in an opportunistic way. This task may involve knowledge in the whole pipeline of ML infrastructure runtime & API. The tech stacks I am working on are C++, gRPC, and Protocol Buffers.

Challenges

1. Testing

In addition to the Unit Testing I was familiar with, I had to do separate Mock Testing on the client and server sides of gRPC, which was a new challenge for me. After some study, I have mastered these methods, and can be said to be proficient in gRPC development.

Apart from this, I also had to do Integration Testing, which in my mind is similar to running all the components inside a real simulated server. However, this was difficult for me, and I didn't have enough time to study it (internal testing tools and internal server configuration that requires 200 lines of configuration to start up a hello world server…) on the timeline, so I didn't finish this part of the project, which left me with some regrets.

2. Performance Benchmarking

I didn't test the performance improvement of the entire library because I ran into a snag with Integration Testing. I ended up testing for theoretical values, but did not plug into an actual server for integration.

I also had to do a lot of extra work to test the potential enhancements of the library on real servers: talking to potential users, asking for permissions, and designing and modifying the library for use. This is something I'm sure will need to be improved upon.

3. Communication

Although we have increased the frequency of weekly meetups since midpoint's evaluation, it still takes a lot of communication to rationalize the specific design of a project and its corresponding implementation. In a code change request, we usually have more than 10 or

even 20 comments to exchange opinions. I feel that this is a more efficient way to communicate directly with the code.

I think the most important thing I learned from this internship is that the most important thing in programming is not writing code or designing code or architectural design, but the communication between team members.

Successes

1. Peer Coding

Since my host felt that my efficiency needed to be improved, we did some peer coding in the last few weeks, and I found that I learned and clarified a lot of his needs for the project in the process.

I think the whole process of peer coding is very efficient, and it also improves the efficiency of the whole communication. I will try this way in my future work.

2. Full Stack Development

My previous technology stack was basically front-end and web development, so the projects in this internship were completely new to me (C++ servers and libraries).

I was not confident in my C++ and server development skills, but I found that there were no major obstacles during this project, so I believe that this internship has made me a step closer to full-stack development. I learned about the development of C++ libraries, the use of various underlying C++ libraries (absl), C++ testing (unit, integration, mock), and the whole process of development.

Other experience during the internship

This is my second time interning at Google. This time my team is fully focusing on the infrastructure and it’s just programming.

On a company level, I don't think my internship experience was any better than last year. For example, Google's cafeteria no longer offers unlimited togo bags, as well as the percentage of return-to-office throughout the office are still the same as they were last year when the covid was at its worst.

On a team level, our group is completely focused on writing code for infrastructure, so basically we don't have the concept of product, but rather code is product. I think, from my personal point of view, I still prefer to do some coding with actual products, so that I can exercise my sensitivity to the users, as well as the interaction experience of the product, and the communication with the product manager and other related communication and exchanges.

From my manager and mentor's point of view, I don't think I like their management style too much. They were less proactive in communicating with me, which led to many periods of time where they felt that my output was not up to expectations, and such criticisms came after these cases happened. I think I could work more efficiently if they had more immediate feedback.

Lastly, due to the downturn in the tech industry as a whole and the fact that they may run out of headcount this year, it was almost impossible for me to get a return offer from Google.

Overall, this internship experience still taught me a lot, both about job and about planning my next steps.