CS 131: Choosing a Language for a flynetting swarm

Mehar Nallamalli 804-769-644

Abstract— **The main goal for this project is to decide a good alternative to c++ for XuFly drone project. In the last project, we used Python and the asyncio module to implement a server herd architecture, so I am assuming that this will be a good language to use because of the ML requirements and because we can handle models that are similar to the server herd. In this report, I will compare Java, Python, and C++ to decide which is the best for an implementation XuFly, keeping in mind there is a ML requirement in the future.**

I. Introduction

We are currently using C++ to build our swarm of flies, but since it is being buggy, and because we need to be able to support some machine learning models, I believe we should explore other languages to implement these features. According to the documentation, TensorRT is built on CUDA and enables to optimize inference for deep learning frameworks which allows the development of tools and technologies while reducing app latency and no hardware limitations.

The main goal of this paper is to decide on a programming language that would be the best implementation for machine-learning algorithms and to rely on TensorFlow. We will look at Python, Java, C++, and Go language.

II. Python

As we know, Python is a high-level language and it provides dynamic typing and automatic system management. This is an advantage over using c++ tensorflow API. Python is also better compared to something like Java because you can have different types of programming architectures such as OOP, or procedural. For working with a lot of drones individually, we would need to employ a type of server herd architecture like we did in the last project, and as we decided, the asyncio module is perfect for it.

III. Python Advantages

According to the documentation, TensorFlow is actually written in C++, so it would seem that keeping the code as C++ would be the most beneficial. But the key point to understand is that the API for tensorflow is meant for Python libraries because it is meant for data processing and data visualization. As mentioned before, another advantage is we do not have to worry about memory leaks and garbage clean up since the garbage collector for python will handle closing connections. It is also dynamic typing, so it will have easy management of TensorFlow objects.

If our XuFly devices are trying to utilize the Coral SOM or Jetson Xavier NX in the future, it is good to switch to Python because there is no need to build models from the group up. The TensorFlow Lite models can be compiled to run on the Edge TPU.

IV. Python Disadvantages

As we know, Python code is interpreted at runtime unlike C++ or Java. This actually hurts performance. We also learned that the GIL (a mutex that protects multiple threads from read/writes at same time) prevents parallelism. Python is also not 100% reliable because of it being dynamically typed, so you may face run-time errors.

V. Java

Java is an Object oriented language that is run on over 3 billion devices. It is versatile and is a language that works a lot with objects. We will look at the pros and cons of switching out tech stack to Java.

VI. JAVA Advantages

Using Java for Coral is very efficient. Coral blocks are low latency, low variance, and they have zero garbage. The building blocks together have low latency and distributed systems help optimize performance.

Java is a platform independent language because Java can be converted into Byte code which can then be deployed on many devices. This is useful for us because if we choose to run our business logic on different hardware devices for our drones, we can easily find devices that support JVM. Specifically, the next gen MobileNetV3. This is an on-device ML model and it relies on AutoML. With large models, the CPU latency decreases. For many devices running on the same network, the Edge TPU that is used on the pixel devices is similar to the Coral products to meet the requirements. So, this is an argument that both Java and Python are good options to switch to.

Both Java and Python are better options than C++ because we want to be flexible with choosing hardware devices for upgrades in the future. These machines can be run with either java byte code or use the provided Python ML models.

An argument for Java over Python is because of Type checking. Python is an interpreted language and is dynamically type checked which means that the variable types are assigned during run time rather than compile time.

Java is compiled and then statically typed for each instance of a variable. This can actually be an advantage for our purposes because more bugs and errors will be caught and will have less type casting issues and will make it more efficient.

Another feature comparison is managing memory issues. Both Python and Java are unlike C++ in the sense that they both have garbage collections and are automatically tracked. In python, the variables are on the heap and in Java the objects are on the heap. Keeping track of reference is a bottle neck because there is more overhead, but Java has a garbage collection method that is efficient. Memory management is abstracted and I as the developer did not have to keep track of references of variables.

Talking about reliability, a strong static typing makes that type checking is only at compile time, so then runtime errors are occurring less than Python.

The final advantage is Java supports multithreading. It is not as powerful as multithreading in Go, but it is still more efficient than Python because of the limitations by the GIL. But that does not mean asyncio is inferior compared to Java. In this project, asyncio is a good candidate. We already talked about this as a disadvantage for Python. Multithreading is good for running and executing tasks, but asynchronous framework is good for handle a lot of networking requests. That is why it doesn't matter that Python is not good at multithreading.

VII. JAVA DISADVANTAGES

Java is unlike Go or Python when it comes to abstractions. Java means we need to be careful about where on the memory they live, their scope, and what type they are. Honestly, the disadvantages of Java are the same as C++, to some extent (except for pointers).

VIII. C++

As for as ranking so far, the number one option would be Python, then Java, then finally keeping it as C++.

IX. C++ ADVANTAGES

The TensorFlow C++ API could be faster as is it closer to the machine than Python and could remove many run-time checks in place of compile-time checks. The C++ API might also be easier to debug as C++ has the type information while Python has no type.

C++ is a highly portable language and is often the language of choice for multi-device, multi-platform app development. C++ is an object-oriented programming language and includes classes, inheritance, polymorphism, data abstraction and encapsulation. C++ has a rich function library. C++ allows exception handling, and function overloading which are not possible in Go.

X. C++ DISADVANTAGES

Pointers in C++ consumes a lot of memory, and that makes the language not efficiently scalable. If a developer does not know what they are doing, they can create wild pointers may cause the system to crash or behave anomalously. C++ gives the user complete control of managing the computer memory using DMA. C++ lacks the feature of a garbage collector to automatically filter out unnecessary data

Another issue with C++ is that threads and multithreading is a new concept for C++. Only recently c++ is capable of supporting lambda functions.

Although object-oriented programming offers a lot of security to the data being handled as compared to other programming languages that are not object-oriented, like C, certain security issues still exist due to the availability of friend functions, global variables and, pointers.

XI. GO ADVANTAGES

Concurrency in Go is very easy to achieve because it is built into the language as a first-class feature.

Go is a very fast and efficient language. It has simple structure and does not have classes or type inheritance. This can be considered as a disadvantage, but it makes it efficient. The entire language is based on functions, so it is very maintainable, and development is much faster.

Since Go is a compiled language, and it is statically typed, developers need to make sure they are not mismatching types, but because it is a typed language, it is a compiled language. It also has automatic memory management which makes concurrency faster and more efficient.

XII. GO DISADVANTAGES

Go is less flexible than the dynamically typed languages and does not have 3rd party modules. This makes it difficult for writing machine learning modules and data intensive processing.
Go also has no generics which means the code you write is not reusable.

XIV. CONCLUSION

Each language above has its own advantages and disadvantages. But based on this project's requirements of having concurrent machine learning models that are deployed on a hardware device, some languages are the better choice.

My suggestion to the executives would be to change the code to Python because it is the best for running on devices and is largely supported by ML and TensorFlow models. Even though Go might be the fastest language and has the best concurrency, it does not support 3rd party modules which are crucial in hardware deployment. Finally, Java is also a really reliable language for this project and it on the most devices and it can be converted to byte code which can run on any JVM machine. It can utilize Coral blocks to build a highly scalable, distributed system for our drones.

Finally, I would say we should switch our code base to either Java or Python because of our requirement to have a swarm for robots to explore unknown environments. This requires ML, and low latency, and low-power CPU and AI accelerators. It is easier to manage these requirements with languages that have models that don't require us to build from the ground up.

XV. CITATIONS

[1] https://ai.googleblog.com/2019/11/introducing-next-generation-on-device.html
[2] https://developer.nvidia.com/tensorrt
[3] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-nx/
[4] https://coral.ai/products/som/
[5] https://www.invensis.net/blog/it/benefits-of-c-c-plus-plus-over-other-programming-languages/
[6] https://pragmacoders.com/blog/multithreading-in-go-a-tutorial