CS 131 HW6 Report

Yining Wang,
University of California, Los Angles

Abstract

This paper aims at investigating the possibility of writing a mobile machine learning application using Flutter as the user interface, in which Dart is the programming language. We will look at the strength and weakness of Flutter and Dart when it comes to its use, flexibility, generality, performance, and reliability. We will also compare Dart to other mainstream languages.

1 Introduction

We are trying to build an application that helps people buy things from garage sales. The old version is web-based, and most of the computation is done in a central server with only the user interface running in users' cell phones.

One of the functions is that the user can take a panorama of the items and price tags on display, and the app will quickly report to user on what the best deals are. The current version does this by shipping images to the central server and having it do all the work, but this is too slow and chews up too much network bandwidth. We are trying to make most of the algorithm to run in the phone.

We are using TensorFlow Lite for running the model on cell phones. We are looking into the Flutter user interface toolkit as one of the technology alternatives, and Flutter apps are written in Dart. a language that can be compiled to the ARM machine code used in most mobile devices, with tflite, a Flutter plugin for the TensorFlow Lite API.

We are referring to Dart 2.7 and Flutter 1.12 (2019-12-11), along with tflite v1.0.5 (2020-02-26) in this investigation.

2 Tensorflow Lite

Machine learning at the edge TensorFlow Lite is designed to make it easy to perform machine learning on devices, "at the edge" of the network, instead of sending data back and forth from a server. For developers, performing machine learning on-device can help improve:

Latency: there's no round-trip to a server

Privacy: no data needs to leave the device

Connectivity: an Internet connection isn't required

Power consumption: network connections are power hungry

Therefore, we think using TensorFlow lite to run the model on the mobile devices is a good idea.

3 Flutter (Dart) and React Native (JavaScript)

Because we are talking about implementing a user interface, naturally one alternative would be React (React Native on mobile devices). We compare Flutter and React Native, as well as Dart and JavaScript to see which framework is better for developing a mobile application’s user interface.

3.1 Use

Both Dart and JavaScript are C-like object-oriented language. Therefore, they would both be fairly easy for programmers with knowledge of any object-oriented language to pick up.

However, JavaScript and React has been popular for a long time, therefore has a significantly larger user community. It would be easy for a React developer to find answers to whatever problem he or she has online already answered by someone. On the contrary, Flutter is relatively new and not really very popular. Therefore, Flutter developers might find themselves stuck in some problems that they can’t really find solution anywhere a lot.

On the other hand, Flutter supports hot reload, which makes it a lot easier for developers to do work with, while react native does not.

3.2 Flexibility

The basic building components for Flutter apps are widgets. Moreover, in other frameworks, we have a different set for views, controllers, and other properties. Each and every widget has a fixed declaration for the UI. Flutter has united object model — the widget. Also, Flutter build its UI from scratch, independent of platform. Therefore, it makes Flutter easier learn, but also makes it less flexible.

3.3 Generality

Because React and JavaScript has been around for a long time, there are way more JavaScript libraries than Dart libraries. Therefore, the generality of JavaScript (React) is a lot better than that of Dart(Flutter).

In terms of platform, both frameworks are cross-platform, meaning they support IOS, Android, and web.

3.4 Performance

Dart include the various compilation types. On mobile devices, it has to compile code Ahead-of-Time (AOT), which gives fast execution and low start-up time; but for development it also compiles in Just-in-Time (JIT). This gives the best of both worlds, as purely AOT compilation makes development slower—once you've tried hot reload, you will not want to go back to AOT compilation during development.

Plus, Dart has a third compile option, which is compile to JavaScript, making it possible to share code between mobile and web applications.

JavaScript, on the other hand, cannot be compiled and uses interpreter instead. It has JIT optimization as well. However, because it cannot be compiled ahead of time, running it on mobile devices could have a longer latency.

,

Overall, I would say the flexibility of compilation of Flutter does give it an edge when it comes to performance.

When it comes to interface smoothness, Flutter (Dart) has an even better edge.

React Native is basically divided into two-parts, i.e JavaScript and Native. In React, the application will run the JavaScript and it will communicate with Native through a bridge. This code is then converted into a native app, for example when we are creating animation which is at 60 frames/sec. Then it will become slower due to the communication between JavaScript and native. In every 60 sec, it has to convert code from JavaScript to native and vice versa. Hence it will end up killing the smoothness of animation. In this application multi-core architectures and applications cannot communicate with the machine.

On the other hand, Flutter provides a lot of flexibility in deciding how to organize and architecture your apps. Flutter does not use any bridge because it uses Dart which can handle animation, painting, gestures, rendering and many more by itself. This boost its performance.

Performance is extremely important for our application because doing the model on the local side already puts enough pressure on the CPU and memory of the mobile device, and a UI with bad performance would make it even slower and impossible to use.

3.5 Reliability

Because React has been around for a significantly longer time, less bugs are likely to happen. Therefore, it is safe to say that it is more reliable.

4 Dart and Other Programming Languages

It is hard to compare Dart to languages like Ocaml, Java, or Python in regards to this application we are trying to build, since they are usually used on very different kinds of applications, and Ocaml, Java, and Python are rarely used to build user interface on a mobile app. However, we can do some simple comparison regarding the programming language aspect of these languages.

4.1 Type Check

Dart is statically typed, just like Java and Ocaml. This gives them the advantage of having compile time type check. Python, on the other hand, is dynamically typed and type check happens at runtime. It gives python the advantage of polymorphism. However, it also means that more bugs could potentially happen at runtime.

4.2 Memory Management

When it comes to memory management, Dart Java, and Ocaml uses a garbage collector implemented using the Mark and Sweep algorithm while Python has the reference count method to do memory management, meaning that it frees the memory if no identifier points to it anymore. The reference count method has the downside that it cannot deal with mutual reference. However, it has the upside that it is easy to implement.

5 Conclusion

Overall, I would say Flutter is good fit for the application we are trying to develop, with its high performance and easy to use features. However, it does have the typical weaknesses of a new framework, like lack of sizable user community and lack of reliability.

Bibliography

David Bolton , The Fall and Rise of Dart, Google's 'JavaScript Killer', https://insights.dice.com/2019/03/27/fall-rise-dart-google-javascript-killer/

Diksha Rana, Google Flutter and Dart, https://dzone.com/articles/google-flutter-and-dart

Marco Bellinaso, Flutter: the good, the bad and the ugly, https://medium.com/asos-techblog/flutter-vs-react-native-for-ios-android-app-development-c41b4e038db9

Tetsuhiro Ueda, Using Flutter in a real-life application with 200k downloads, https://medium.com/google-developer-experts/using-flutter-in-a-real-life-application-with-100k-downloads-ab64ecd8e03f