CS CAPSTONE REQUIREMENTS DOCUMENT

NOVEMBER 14, 2017

CDK DATA STREAM AI

PREPARED FOR

CDK GLOBAL

CHRIS SMITH

PREPARED BY

GROUP 65

SIGFIND

JACOB GEDDINGS

CONTENTS

1 Overview

This portion of the document will go into a technical review of three potential options for OpenCV supporting languages. The options are C++, Python, and Java; the pros and cons of each will be noted. Once all options are considered, there will be a discussion that directly compares these three choices. Lastly, a conclusion will be drawn from these three choices regarding the leader that we will utilize going forward.

2 Criteria

Our project has a hard requirement that the AI be open sourced. As such, we are restricted to utilizing established AI platforms. With this in mind, OpenCV is the leading candidate primarily due to its flexibility and popularity when wanting to handle image processing. When factoring in that OpenCV is our AI of choice, there are only three language options that are viable: C++, Python, and Java. Thus at present our only feasible options for supporting the AI are to choose between these three programming languages.

3 Options

3.1 C++

C++ is the primary language used by OpenCV and the most robust in general. C++ is widely regarded as one of the most efficient languages when it comes to runtime resources. It also carries the largest selection of tutorials and guides to use for picking up and using OpenCV. Given the time limit in which we must learn and utilize this language, the complexity is a more important factor to consider than normal. The library supported under C++ is massive when compared to most languages in general, with the added bonus that these are going to be some of the more optimized options for runtime performance. C++ is also very versatile when planning to port it into a wide array of devices. Lastly, the community backing it is extensive. This option has been available for years, and thanks to its status as free-to-use and its efficiency, many users have taken to working with it. As a result, many questions have been answered already regarding the usage of OpenCV within a C++ language. This language does carry problems however. The biggest obstacle is the innate difficulty of using C++. While normally something that can be brute forced to make work, this too is bogged down by another flaw: poor documentation. Complaints about poor documentation and lack of explanation or illustration is widespread. It is common advice to avoid using this language as a first choice when trying to learn it the first time. Another complaint is that while it supports a strong library, its actual pool of machine learning components is rather limited by comparison. This results in difficulties when attempting to use multiple different learning routines on a program. Another issue is the challenge of interpreting code and the debugging process. While this is a normal issue for any programming language, for C++ it is compounded when attempting to make use of machine learning algorithms. If the programmer needs to write any code from scratch, this method becomes a nightmare for comprehension. Other factors to consider are the teams individual capabilities when working in this environment, primarily considering our background experience, preferences, and capacity to work in this environment. Our background matches this language the best given that Oregon State University has a strong preference towards teaching C++ as an entry level language, but consideration has been made that this requirement rapidly relaxes after the first couple terms in the course. Preferences

are where things become troublesome with C++ with the acknowledgement that no member of the group is actively supporting the adoption of this method; Python and Java are more popular selections. Lastly, the ability to work in this environment may be taxing on our team. The time it takes to effectively learn this platform/language may be too great for us to complete the project within the prescribed amount of time. Fortunately, in this final note we do have the assistance of a teaching assistant that has specifically operated on OpenCV in C++ as an immediate resource for us to understand how to properly use this program.

3.2 Python

Python is the leading rival of C++ in language preferences. The most popular strength of Python is the simple ease of use associated with it. The language is straightforward and easy to write in. This ease of use is also greatly assisted by the short and concise nature of Python. Code is in general praised as quick and easy to implement as well as comprehend. Another interesting fact about Python is that its often regarded as a language of scientific computing, with support for packages such as OpenCV, numpy, scipy, scikit-learn, and matplotlib[1]. In direct contrast to C++, the debugging and visualization of Python is considered very good which is primarily the result of its previously mentioned simple-to-understand syntax. Also, portability is something that should be considered with our program, and as such popularity and use of Python means that most systems already have support for this language. Beyond these strengths lies a few weaknesses however, mainly in the form of weak documentation, inferior online support, poor performance, and the cross-use of languages. The documentation has made great strides to catch up with the primary C++ language, but it is still considered trailing which means the usual concerns arise. These concerns include difficulty discerning what a function does on a deeper level, what the various parameters do to impact the program, and relatively weak guides in the usage of this version. The weak guides portion cannot be understated either; our team needs to learn this platform and execute its construction relatively quickly, and that is not a likely goal if we are faced with insufficient resources off of which to help build our code. Another issue is the cross-language usage that will be present. While a distinctly Python language, it does make use of libraries such as numpy. This means that C++ is constantly being utilized in the background. This isnt an issue unless we need to dissect a function in full or construct a new one. Should that event occur, well be faced with the issue of juggling multiple languages at once as opposed to a single one. As for team support when considering backgrounds, preferences, and flexibility, this language does have a bit of support. While not explicitly required by Oregon State University, Python is a popular choice amongst students for completing projects, and as such all members have prior experience with this language. When factoring in our preferences, this is the language with the most upfront support. Lastly, this language runs into an issue where the poor documentation is concerning, but also our leading resource - our teaching assistant - does not have extensive experience in using OpenCV on Python. As such, comprehension will fall more on us. These is also consideration being made that should OpenCV completely fail us in our task and we attempt to move over to a separate open sourced AI, then Python would be a strong candidate to already be working in given the probability well favor a Python based AI such as TensorFlow.

3.3 Java

Java is the final language being considered for use within OpenCV. Java on its own is the leading competition to C++ in regards to object oriented programming. Often a popular choice for being an easier to implement language than

C++. To gain this advantage it does carry the added weight of being less optimized during runtime with performances generally trailing behind that of its competition. Beyond these two points the two languages are on relatively even footing outside the context of using OpenCV, once considered however, the differences between the two languages becomes a pretty significant problem. This language as a choice is a bit different from others being considered in that, while all lack documentation, this one takes it to the furthest extreme. Having not existed until version 2.4.4 of OpenCV, Javas introduction is the latest of the bunch. Its primary focus is on its usage in Android devices[2]. When attempting to research this language in particular, there was very little mention of it at all, with tutorials and guides being few and far between. While Java itself is a reliable enough language, it has failed to gain traction when placed in an OpenCV environment primarily due to its middle-of-the-ground nature. What this means is that its runtime performance is generally in-between C++ and Python. This means if performance was a concern, then C++ is the better option. When considering ease of coding, then Python wins, with Java falling into second place for either category. This is only made worse by the previously mentioned lack of documentation, which means it is extremely difficult to make proper use of this language. While other languages also suffer from this, they at least have several years of exposure in which the documentation has been improved or added. Java on the other hand has had minimal additions. When factoring in group experience, preference, and adaptability, the situation doesnt improve by much. While Oregon State University does instruct students to use Java, it wasnt taught to the same degree as C++, and it is not a common choice amongst students when picking languages. When it comes to our group preferences, we do have a vote for the use of Java, but given the challenges associated with this platform it is a cautious vote. Lastly, as our primary resource for getting over obstacles, our teaching assistant has not even mentioned the use of Java as an option, let alone indicated that he has worked in it before. This coupled with non-existent online support leaves Java in a tricky place for consideration. It does carry the merit that, should we back out of OpenCV and move to a different platform, there is a popular Java focused AI called DeepLearning4j. As a result, operating in the same languages while changing AI platforms would make it easier to transition, but this would be preparing for failure. As such, this is not a factor that can be used as a major point of value and instead more used as a reminder of what can be done should this current plan of attack fail.

4 Discussion

These three choices are largely different from one another with the only considerable common ground being that they all have poor documentation in general. The discrepancy however, is that of the three options C++ does carry the strongest online support and Java is at the bottom by a significant margin. When factoring in the assistance that may be provided by our teaching assistant, the argument to use C++ only strengthens given his prior exposure to it. Next for consideration is ease of use for coding purposes. In this aspect, Python is the strongest candidate and Java and C++ are the ones trailing behind. When considering code efficiency at runtime, then C++ is the clear winner. Lastly, as far as team preferences go, the most voted choice was Python as a general language preference, however, not when considering OpenCV. In this vote, Java followed up in second and C++ was the least popular.

5 Conclusion

Given what is known about the three choices at present, the current leader that will be selected is C++. We are confident that with the support of our teaching assistant we will be able to circumvent the obscurity surrounding its documentation

and be able to take advantage of its optimization and flexibility. There is also the possibility that since Python utilizes C++ under the hood, should we need to try and move to a different language due to library restrictions the migration wouldnt be too taxing as opposed to involving C++ or going from Python to C++.

6 References

[1] ”LearnOpenCV” Internet: https://www.learnopencv.com, Oct. 30, 2015 [Nov. 14, 2017].

[2] ”Introduction to Java Development” Internet: https://docs.opencv.org, Nov. 13, 2017 [Nov. 14, 2017].