6. Project Management

6.1 – Project Plan

As I am the only member of this team, task responsibilities as well as contributions are 100% done by myself. Because of this, I must keep my pace quick in regards to development/testing. My plan is to always have a stable build at each project report stage, and to make substantial feature implementation that builds on one another, as opposed to having disparate features that are poorly actualized. By the second iteration, I am hoping to have integrated the Spark Server fully with the existing Android code, and to then have an object formed that would be ready for a database update. For Iteration 3, I would expand the application to include the database relationship and begin work on the Google Charting API functionality. On the final iteration, I am planning to have metrics displayed on the web portal based on the Charting API work done in iteration 3. This would also have the final completed algorithm for the question answering as well as any neural network tweaks that need to be done as the course progresses.

6.2 – Work Completed

So far, I have completed the Android user interface and provided the entry points to the application’s main functionality. I was able to implement an algorithm that completes the Skip button’s functionality so now the remaining workflow focus is on the ‘Ask Question’ feature. On this front, I was able to add the API code to launch the Google Speech Recognition prompt, and to then store that recognized speech as a string variable. As a means for testing, I have also added the code required for the application to verbally speak back to the user. Code has also been added that links the Spark Server, which was also designed, to the Android application; albeit currently with only GET functionality tested.

For Iteration #2, I changed the UI to include a redesign of the home page and buttons. By making both buttons larger and red, they are now more pronounced and friendlier to the user. I also replaced the Skip button’s shuffle functionality, which only shuffled between pre-existing images, to a user gallery selection so that any image can be used for the application. Because of this lack of a default pre-existing image, I added a home screen that prompts the user with app instructions and waits for their input. During this iteration I also added the Clarifai functionality so that any image can be passed from the Android app to the API in order to receive image recognition results. As a result of this, I added the functionality for the application to speak the answer back to the user so that the interaction is more conversational in nature. Finally, I added error handling to properly route any issues experienced with the app itself.

For the final iteration, I worked on a CNN tensorflow model based on the the COCO and DAQUAR datasets. This allowed me to pass an image and a question, and to then receive an answer with a 50% accuracy rate. After several attempts at trying other models but running into Ubuntu or Windows compatibility issues I went with a tensorflow model running on Windows 10 via python scripting. In order to leverage this from Android, I had to use Heroku to host the python app so that I was able to pass parameters into it for my image analysis. In this iteration I also added the metrics functionality which now exists within the android app itself (top-right corner). I also created some initial charting that can be seen on the UI that draws from the SQLite database which was also finalized in this iteration.

6.3 – Pending Work

I would like to add more charts as only a few options are currently available. I would also like to add deeper semantic analysis which could be based off the charting to further analyze the frequency of certain question types in relation to their cognitive visual recognition level. I would also like to eventually move the tensorflow processing itself to the android device as that has been done in recent months.

6.4 – Issues/Concerns

As mentioned above, I would like to move the tensorflow processing entirely to Android. I made multiple attempts to do this but hit various configuration issues. Given more time, I would like to implement this wholly on Android. Another concern that I had was with questions that are specific in nature. I’d like to grow this model so that it handles these questions more accurately as it will currently only provide a vague response if the question is asking too much from the model.