Attention Detection using Google Cloud Platform

Summary of Project Progress:

Sabra: The goal of this project was to create a method of detecting if someone in a picture is paying attention to a screen or ignoring it. The aim was also to use a built-in algorithm on the Google Cloud Platform.

At the beginning of this project I researched how others have attempted to create gaze tracking. Since trying to gauge whether a person is paying attention to a screen is very important to large corporations, there isn't a lot of information about specifically how people decided to create gaze locking. It was also difficult to find a dataset with appropriate labels. Eventually, I did find a dataset of almost 6,000 images called the Columbia Gaze dataset. This dataset has labels that include vertical and horizontal head positions and gaze directions. I also had to determine the criteria of which combination of head and gaze positions would be labeled as "looking" versus "ignoring". Finally I found a criteria that I felt comfortable with, but the classes are unbalanced with the "ignoring" class having over 4,000 images and the "looking" class only having over 1,000.

I first attempted to use GCP's Vision AutoML image classification. Although, the results were poor, so I turned to another algorithm that would give me slightly more control over the model. GCP has a built-in image classification algorithm on their AI Platform which allows you to hyperparameter tune your built-in algorithm. The model, however, requires that the dataset images be in tensorflow record format. I tried following the tutorial on how to convert the image files in GCP cloudshell. However, there were several issues with the tutorial as it was out of date. I finally had to create a dockerized container to run the tensorflow conversion on my machine.

It took several attempts, but I finally ran a semi-successful training job on the AI Platform. For this I hyperparameter tuned the final model, giving it the maximum number of trials, the widest range for weight decay and learning rate, and finally used efficientnet-b4 which is a state of the art image classification model. The result was a model that did OK at predicting "ignoring" and a not very good job of predicting "looking". However, this was much better than previous results. For a better model I believe more data would be required as neural networks work better with more data.

Flask Development:

Melissa: I was responsible for developing the Flask app. This Flask app consisted of two separate pages - *Attention Detection: Image Selection* as well as *Attention Detection: Model Results*.The Image Selection page needed to house two different features - the ability to take your own picture with your webcam and the ability to upload a picture. These pictures had to be sent to the model, and results needed to be displayed on the Model Results page.

While I had the generic structure of a flask app from the week six homework assignment, I still had to create a function that allows the user to take a picture. Plus, we wanted the Flask app to be more aesthetically pleasing than just plain HTML. I

started by tackling the camera feature. It was surprisingly difficult to find a python library that not only opens your webcam and takes a picture, but is also supported by Windows, Linux, and MacOS. Fortunately, OpenCV worked well and was (supposedly) supported by all three operating systems.

After creating the camera function, I had to figure out how to integrate this function into the Flask app. This was fortunately rather simple; I just created a button that triggered the camera function when pressed. Once this button was created, I had a rather plain but fully functioning Flask app. To make the Flask app more visually appealing, I edited the HTML to add colors, dividers, headers, and bolded text. I also used CSS to make the buttons larger and blue.

Regarding detours, I had a handful of obstacles when developing the flask app. Primarily, making a function that recognizes and launches your webcam, takes a picture after a certain key is pressed, and sends this picture to another function proved to be rather difficult. I tried a handful of libraries before getting the OpenCV library to work, and I spent a lot of time researching and debugging. After I got the function working correctly, it wouldn't work on Sabra's computer, which was another obstacle. Where I have a windows computer and she has a Macbook, it was difficult to make the function work on both of our operating systems. We also realized the camera function didn't work in the docker container, and pressing the triggering button crashed the website. Although I was unable to resolve the issue myself, Sabra was fortunately able to edit the function and have it work correctly in the docker container. I also struggled with editing the "choose file" button. For some reason, you can't style an input type="file" button easily with CSS like you can with any other button, and I had to import webkit-file-upload-button to edit it.

Sabra: The open camera function we had originally created only worked on local machines. If we ran it in a container then it failed. So we found resources to help us build a page with JavaScript that accesses the users camera locally. It also builds a separate page that lets you take a photo, select your photo, and submit it to the model.

Dockerized Container and URL: Sabra

I created a base ubuntu dockerized container within a home PC. I placed our codebase and our flask app within this container. This is hosted on a private cloud, and I used the default port 5000 so people can access the app externally after I confirmed it with the firewall. I also have a line that I run in the dockers terminal to have the app run constantly.

Your final amount of progress (and how it compares to your safe/stretch goals)

Sabra: Our progress is exactly what I would hope to do with this project. However, I think that it could obviously be improved such as the model performing slightly better with more data. Another future goal would be to track someone's gaze during a live video feed.

Melissa: I am very satisfied with how the Flask app looks. Not only did I learn a lot about HTML and CSS, but the Flask app looks even better than I expected. However, I

was never able to resolve the camera function issues, which was both frustrating and disappointing. My camera function works perfectly on my computer in a virtual environment, but did not work on Sabra's computer or in the docker container. This was an obstacle I didn't anticipate facing, since the Python library I used claimed it worked on both Windows and MacOS. I believe if I had more time I could have resolved this problem, but I was unfortunately unable to get it running perfectly. Thankfully, Sabra had the time to edit the function and figure out the issue.

To address my goals, the aesthetic of the Flask app exceeded my expectations. Regarding the camera function, I reached my safe goal of creating a function that at least works locally, but I was unable to reach my stretch goal of implementing this function within the docker container.

Things you found helpful and things you found unhelpful along the way

Sabra: I found the GCP tutorials useful as a helpful guide. They were also not very helpful because some were out of date and some others described algorithms that were in beta. The AI Platform's image classification built-in algorithm is technically in beta testing as well.

Melissa: I used Open-CV python tutorials when developing the camera app, and it was super helpful.When I was editing the HTML and adding CSS to the Flask app, I really benefited from online tutorials as well. Of course, like every programmer, I also benefited from StackOverflow and GitHub discussions, especially when working with confusing bugs and error codes. Regarding things that were not helpful, I don't think I encountered any resources that were not beneficial, which I'm grateful for.

A summary of the lessons learned from working on the project; particularly about teamwork and communication.

Sabra: I learned a lot about finding and preparing data which is technically the most important part of data analytics. I would also say that having a good project partner who is willing to put in the same amount of energy into a project is critical. Melissa and I worked very well together. We communicated about the project frequently, unfortunately finals week is busy for everyone but we pushed through.

Melissa: I enjoyed this project a lot. I never had the chance to work with HTML or CSS prior to this project, so I was happy to develop the Flask app. Working with Sabra was great; we communicated almost daily, met frequently via Zoom, and were always on the same page. After working with Sabra, I realized how important it is to have a good project partner. Because of our effective dynamic, we got a lot done in not a lot of time, and we were able to develop a decent program. Although, I wish I had more time to work on this project, because I experienced a lot of obstacles and errors that made progress slower than predicted. Considering this, I have learned to always anticipate programming to take much longer than expected, and anticipate facing many errors and obstacles. Similarly, I have learned to prioritize choosing a good project partner, because it was wonderful working with Sabra.

If you could do it again, is there anything that you would either do differently or change about your project idea itself?

Sabra: The project itself I wouldn't change. Overall I felt like my approaches to finding the data and creating a model to be the most appropriate path. I think one thing I might have wanted to do is try building a model from scratch instead of using just the built-in algorithms. I think I could have more efficiently tuned the model and gotten better results.

Melissa: If I could do this again, I would have approached the camera function differently. Neither of us anticipated the camera function being so problematic, so discovering that it didn't work was an unpleasant surprise rather close to the deadline. If I could do that differently, I would have had Sabra help me with developing the camera function from the beginning, so we could've made sure the function worked on both of our computers every step of the way.