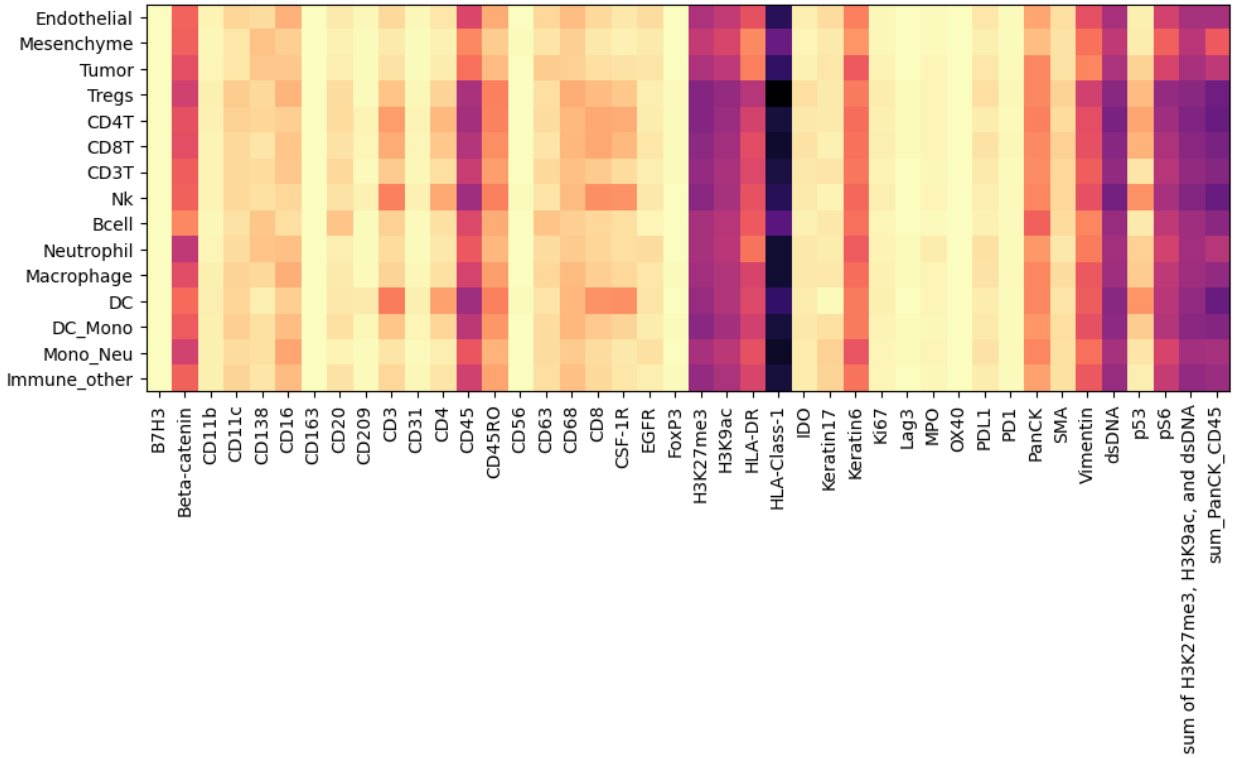
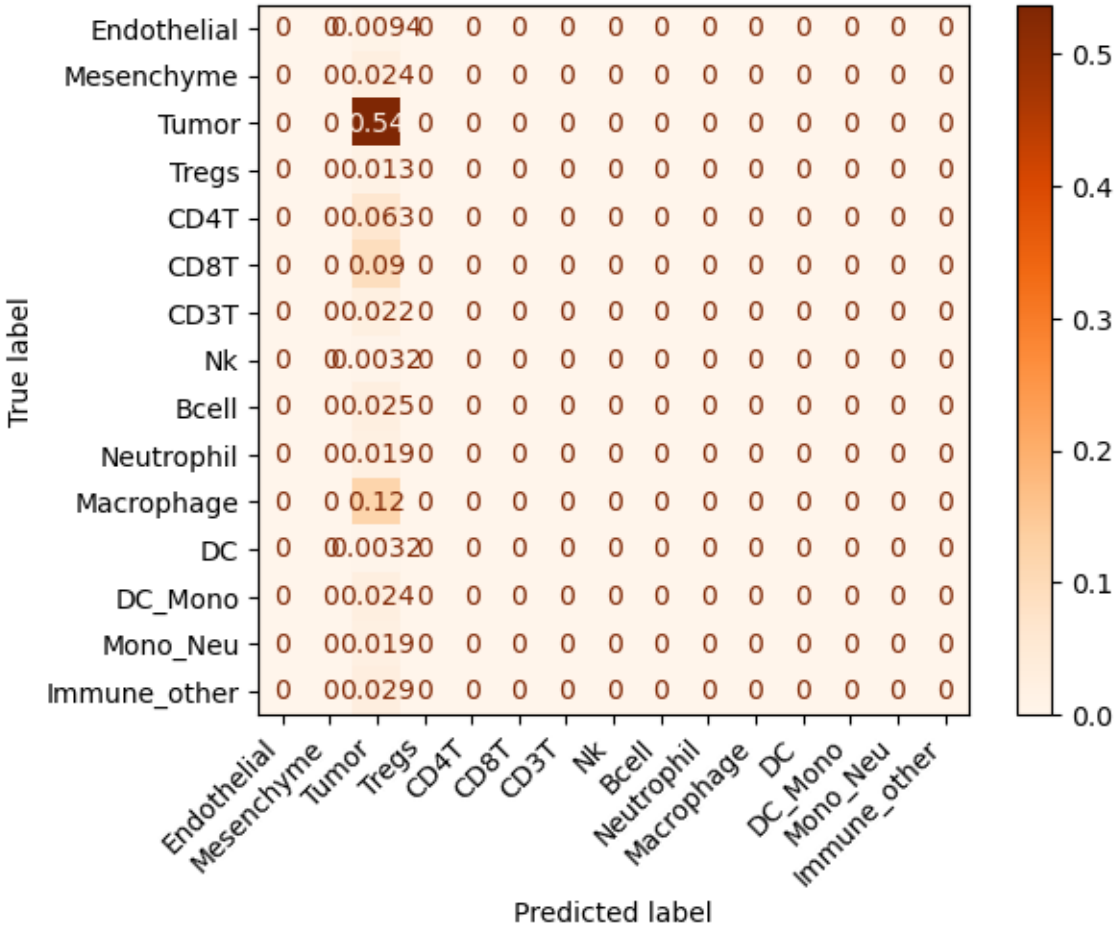
BE/Bi 205: HW1

Danny Collinson

1.



2.



3. https://github.com/DannyCollinson/BE205/tree/main/HW1

4. In preprocessing, I used non-adaptive histogram equalization because it is easier to implement than adaptive histogram equalization but still has the desired effect of enhancing the contrast in the images. I eliminated the Background, Unidentified, and Failed Harmonization cell types because they are not relevant in searching for marker genes, and I also removed the non-biological channels for the same reason. I only used half of the data files because processing them took a long time. For each cell, I took the smallest bounding rectangle of the whole cell mask and calculated the total intensity in the cell for each channel, then divided by the volume of the cell, then averaged across all cells of that type to make the

marker expression panel. I did not save the images of each cell, only the sums and averages for each channel because of memory issues. I made Pytorch datasets and dataloaders that had one-hot encoded versions of the cell type labels and the channel sums normalized by the maximum sum in that channel, then passed that data to my model, which was a neural network with an input layer with 42 inputs (the normalized sum for each channel), three hidden linear layers with sizes 34, 27, and 21 to condense the representation of the data as it progresses to the final output layer of size 17, one for each class, where each hidden layer also had dropout with p=0.5 to help regularize the model, and a ReLU activation function since this is usually a safe choice. The model's prediction is the class with the highest value in the output layer, which we compare with the true class to get the accuracy and the confusion matrix. In the confusion matrix, we see that the model exclusively predicted "Tumor", which is because over half of the data I used was from tumor cells, making the strategy of always picking "Tumor" a good way to minimize the error. This could possibly be dealt with by resampling from the underrepresented classes to get a more even distribution of training data, but this likely wouldn't overcome the relative lack of data I was using since the spatial information had all been condensed into summations only. If the Google Cloud was working for me, I would have tried running a CNN on the full actual cell images to see how the performance may improve, but I was having technical problems and was unable to run it on the Google Cloud environment.