Nick Lockett
ECE 590-6 CV and ML
10/15/17

Individual Contributions

The goal of this course is to develop a program to identify and locate a threat brought through a TSA security machine. The course was divided into two groups, and I was assigned to work on classifying the existence of threats. Although I'm not sure I remember everything I have worked on, my main contributions towards this end fall into the three distinct categories of feature detection, general infrastructure, and Deep Learning Development.

Our original approach to classification consisted of developing a set of features informed by our own heuristics. We planned to calculate several values to assess the symmetry, brightness distribution, closeness to a circle, and many others and then combine them through an SVM or other suitable ML algorithm. I worked with another student to build out the SVM and the symmetry detection algorithm using several python packages. The Symmetry detection was applied within a single slice of the scan, and looked within one "limb" to consider internal symmetry, as opposed to comparing one leg to the other. We used open CV to match circles to the shape, and then look at each mirrored half. Unfortunately, the feature detection proved to be less effective than we imagined, and we moved on to other strategies.

Almost immediately we recognized that we needed a way to access smaller pieces of the TSA's scan. This piece of infrastructure would be crucial for our future development. After meeting up as a group and planning out an algorithm, I wrote the code to search through the a3d file and extract a cube with all of the necessary information. Specifically, the code used a threshold value to find the edge of the actual body, and then took a cube of N size from the surrounding area. It also saved some key information, such as what region of the body the cube was from and whether it had a threat or not.

Once we had this scan we decided to begin moving forward with a more advanced method of analysis. I spent several days researching, and selected a few academic and reputable Deep Neural Networks designed for image recognition. Since implementations of these architectures were readily available online, I opted to adjust one to our uses rather than attempt to re-implement my own from scratch. Specifically, I had to adjust the sizes of the matrices in the convolution layers to take in a cube of values rather than a 2 dimensional triple of values usually associated with an image, while ensuring the intent of the convolution was still accomplished. This was fairly easily completed, and now I am awaiting better data to begin training. Towards that end, I also built out a script to take in the output of imageJ threat marking and combine the results of the into a single, easily usable file indicating the threat.