

Enhancer Activity Detection Hyperparameter Tuning

By: Ben Halligan

# Research Goal

This research is intended to optimize several convolutional networks by tuning the hyperparameters to the specific human data set we have. In order to do this efficiently it will be separated into batch jobs to be run simultaneously.

# Background

In our previous research, we compared complex multilayer convolutional neural networks to a simple single layered convolutional neural network using a relatively small data set. From these comparisons we found our small single layered CNN out performed all of the complex CNNs and we believe that in order for the more complex CNNs to do better we would need to tune them to our dataset as well as increase the size of the data set itself.

# Project Plan

The first step would be to convert the existing CNNs Juypter notebooks into Python scripts to be executed in batches. Then we would create sets of different hyperparameters to test for the highest accuracy results. From these results we would combine the best performing hyperparameters to achieve the highest possible accuracy. The process of determining best hyperparameters would be done using batch jobs of several different settings such as dropout value (i.e 0.1, 0.2, etc.) and comparing the outputs of the different models or using a greedy algorithm to automatically select best case hyperparameters.

# Results

I started by converting the Jupyter notebooks into python libraries for calling from a main file. In the main file itself I imported the one hot encoding and function to import and split the data. With the switch to online classes and the leaving of Rosie’s manager Dr Nowling made the switch to his personal cluster for the project. I was not able to create the proper python virtual environment for the batch program due to not having root access. With the increased difficulty from online classes and changing Senior Design requirements I wasn’t able to keep up with the research project. Of the work I was able to complete was a function that takes a built model, the loss, optimizer, and result file name and runs the learning process and records the results to the test file. Overall, the rapid shift to a much harder class system led to poor time management on my part.