Current State of the Project:

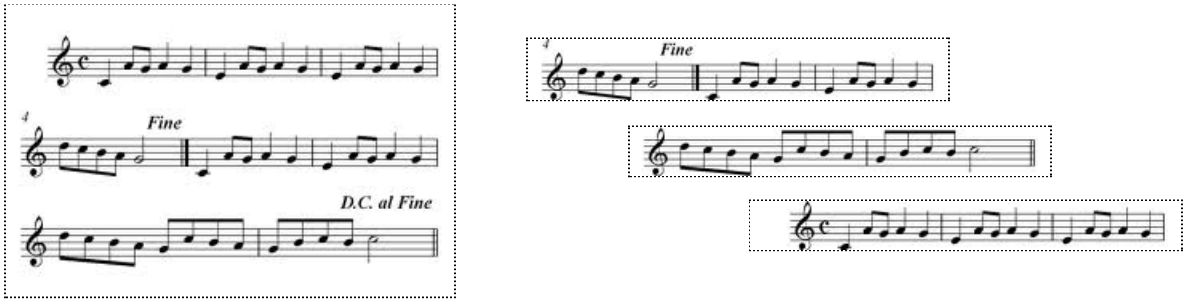
*****For our latest code, please check out the milestone 3 branch.**

Classification Model & Dataset:

This milestone, we gathered more datasets to train on our model. These datasets contain multiple staff-lines (not only monophonic) with the same ground-truth label format as the PrIMus dataset. The data was all retrieved from this website—http://kern.ccarh.org/help/data/—where it has a collection of 10,000 monophonic scores. However, many of these scores are in a unique format called "humdrum (.krn)." We developed an automated script that converts all these humdrum files into an xml file, which then converts it into png format and extracts all the note attributes which are used for our ground-truth labels. The dataset can be found here: https://drive.google.com/drive/folders/1GjcAfxccS6Ao8h3_7lwgYYmIhhQ8IzS_?usp=sharing.

We also created an additional dataset, as previously mentioned in milestone 2, that split all the individual notes of each music score in the PrIMus dataset, so that our classification model be trained using each distinctive note.

When an individual happens to send in a music with multiple staff lines, it can now be splitted into multiple individual staff lines, which will be then splitted into music notes and fed into the model for training/classification purposes. The function can process slightly blurry images by changing the variable of "devi", and the image with one color channel by setting the variable of "oneChannel" to true. An example of the splitting process is shown below.



Mobile App:

Currently, the application has 2 additional functions - one is sending a basic http get request, which was created for testing purposes and was part of the learning process of setting up the client-server interaction. The second function is the actual uploading of a selected image (that is obtained from using either the camera or the photo gallery of the device) to the server. Also, the issues that happen with accessing a device's photo gallery have been fixed.

Server and Back End:

The virtual machine is currently hosting an Apache server that can be accessed using the link: http://35.239.235.42. The server has a few test pages such as /page that allows communication of data between the phone application and the web server (simple data such as strings).

What tasks were done:

- **John**: Created a new dataset of over 6,000 images and developed the automated script that converts humdrum files into xml and png formatted data and outputs a ground-truth label. Trained classifier model on these datasets.
- **Jefferson**: Wrote the API on the android application necessary to communicate with the server. Currently it can serve basic http GET requests, and the code for sending an image through a stream of bytes via a http POST request has been completed.

- **Ying Qi**: Reconfigured the Apache server and allowed the Flask development framework to be implemented on it. Also created a tentative web page that responds to GET and POST requests that allows rudimentary communication from the app to the server through internet requests. The web page can also process simple data that is sent from the phone application to the server.
- **Houlin He**: New dataset is found, and figured out a way to convert HTML files into PNG files to extend the dataset. Created a script so that a music score with multiple staff lines can be properly splitted horizontally.

Proposal Changes

We only had 1 minor change—we did not create a dataset with chords. The way our classifier and ground-truth label is formatted cannot output chords and is restricted to only classifying monophonic scores. Because our model is based on the research paper—End-to-End Neural Optical Music Recognition of **Monophonic Scores** by Jorge Calvo-Zaragoza et al—there is a lack of further research being conducted using the CRNN model to classify chords.

Current Milestone 3:

- Create new datasets with similar images and ground-truth labels, mix this in with our current dataset and train it with our model.
- On the android application side, the API for interacting with the server has been implemented
- Handle musical scores that have multiple staff-lines.

Current Challenges

Classification model:

- The model always outputs the notes a half-step off (i.e Outputting a note as sharp when it should be natural).

Android App and User Interface:

- While the code for uploading an image to the server is complete, it has undergone little testing since it is reliant on the backend.
- While the application can receive simple plaintext and json format responses from http requests, the capacity of the application to receive significant information from a server (such as an MIDI audio file) and handle them correctly is still uncertain.

Server and Back End:

- The challenge remains on how to properly add the machine learning model to the back and server. This is already made easier by the decision to use Google Cloud Platform.
- Another challenge is how to properly interpret the bit stream sent from the phone application as an image.