CS 6170 - Project Progress Report

Yash Gangrade (u1143811) and Siddhant Ranade (u1141131) April 2, 2019

1 High-level Project Description

In this project, we are exploring the topological structures and feature in the Music data. Music in general is rich in structure. We are looking at similarity measures and distances between single notes, sequence of notes (ordered), beats etc. and we are looking at the topological structures present in these. We extract these topological structures and features for different types of music data and we want to apply them for getting results of the higher classification tasks like genre classification, composer identification, comparison of chord progression and so on.

2 List of completed milestones

- Till this point, we have completed the data collection which includes downloading the required data, selecting the channels for initial milestones by extracting the features from the data, manually labeling and selecting the files.
- Next, we have converted the channel data to point cloud form based on the type of feature we want to focus on. For example, we can create a single note point cloud from the MIDI data or we can create a point cloud for ordered sequence of notes (like takens embedding).
- Once we have the point cloud, we compute the internal distances of point cloud which will assist us in using ripser for computation of persistence diagrams/barcodes.
- Then, we apply distance measures like bottleneck or wassertian distances to compute the similarity or the distances between these diagrams. After that to verify, we can plot them using any dimensionality reduction technique i.e. TSNE.

3 List of upcoming milestones

- Trying out simple benchmarks from the previous work i.e. [xxxxxx] next week.
- Use machine learning algorithms along with neural networks for the artist and genre classification based on the songs. We will compare it to the state of the art as discussed in [xxxxxxxxxxxxxx].
- Include multiple channels of a song to perform the classification technique as aforementioned. If this works well, then our project could be generalized to any general midi file.
- Identifying the common chord progressions in popular music (popular music basically uses 4 chords transposed to whatever key the song is in – can we identify these?). Also, we do want to compare it with the classical sonatas.

4 Preliminary results

The results and procedure for the current milestones is discussed in different sections as follows.

4.1 Data Collection and Processing

The data collection for us has been more of manual labor where we find the data and extract channels from it manually. As of right now, we are using 30 midi files to show our work but we plan to use about 100-120 midi files to achieve the rest of the milestones. These 27 midi files have 3 different artists namely Queen (10 songs), Coldplay (15 songs), and Red Hot Chilli Peppers (5 songs).

- Download the midi files for different type of songs, for e.g., pop, rock, classical etc. from any of the online source like bitmidi.com. We have used songs from Queen, Coldplay etc. bands along with solo artist songs like Taylor Swift, as well as we have used classical sonatas like Beethoven, Fur elise etc. Having the diversity in the data provides us new insights and also help us to prove the correctness of the project. Now we have the raw data for our project.

- Use a open-source software called MuseScore (https://musescore.org/) to manually select a channel from the midi files and then exporting it as another midi file. Channel selection is based on the information each channel provides, usually for topological data analysis, piano, drums, guitars are considered good channels because they have some patterns we can find. At this point, we are selecting one channel per midi file but for the future milestones, we will use multiple channels for a midi file. Please find the attached screenshot of how the channel selection works in the software.

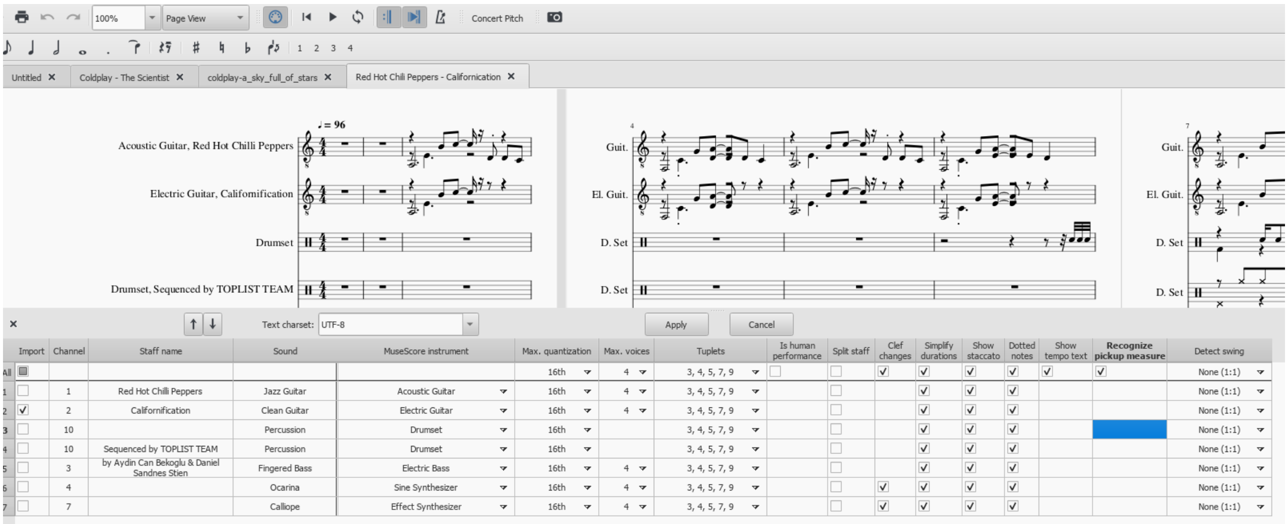

Figure 1: Channel Selection in MuseScore

4.2 Extracting point cloud and analyzing persistence diagrams from the data

- Read the file using mido library in python. Then we have implemented some helper functions to create point cloud from the extracted data. One of the functions we have implemented makes a list of lists in python where each element is the timestamp in the midi file along with the midi message. Midi message is just the information about which note was pressed on which channel with what velocity (eg. pressure applied on piano key) for what amount of time. Then, finally we implemented a function which preserves the time series while creating the point cloud. Essentially we follow takens embedding to create this point cloud. We decide the number of dimensions we want in the point cloud, for eg. with 3 we will get *(note1, note2, note3); (note2, note3, note4)* and so on.

- Once we have the point cloud, we use another helper function to compute the distances between the points in the point cloud. Here, we also use the fact that the notes are cyclic at every 12 notes difference. Then we have a distances matrix which we pass along to the ripser to compute the persistence diagrams.

- We perform the aforementioned procedure for all the midi files and store all the persistence diagram in form of a list. Once we have this data, we compute all the bottleneck and wassertian distances for all pair of persistence diagrams. This part is similar to what we did for project 2. Finally, we plot the distances after applying the dimensionality reduction techniques specifically TSNE because it works well in preserving the distances.

- Please find the results in form of the TSNE plots over Wasserstein and Bottleneck Distances in the subsequent page.

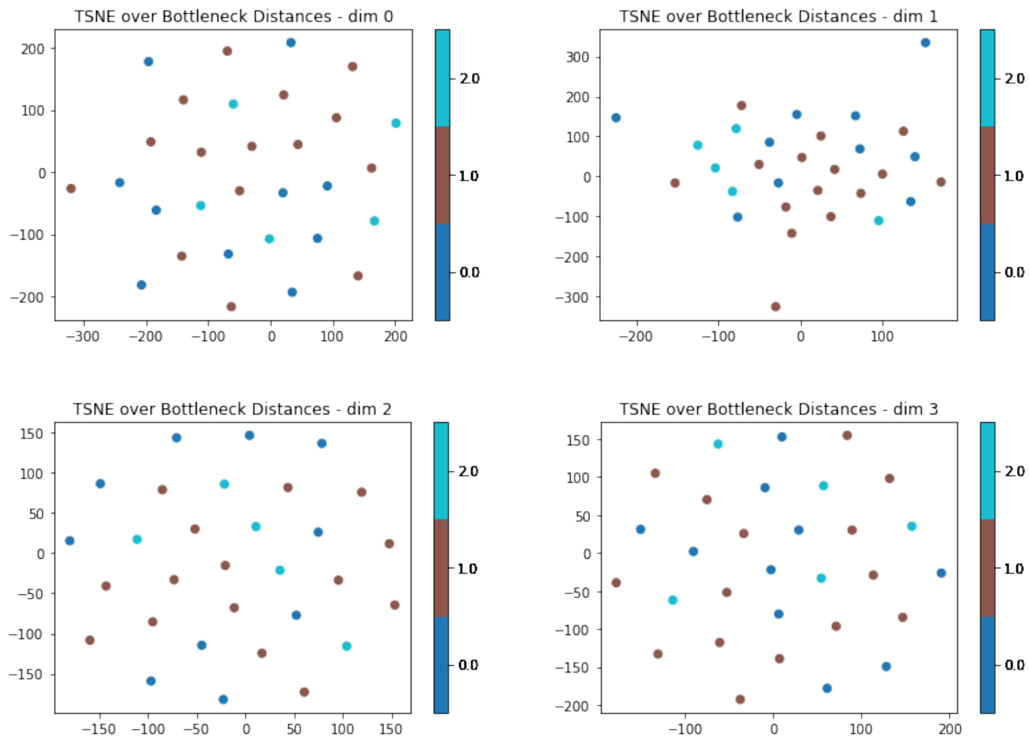

Figure 2: TSNE applied to Bottleneck distances over persistence diagrams. Here, the number of files = 30 with 3 different artists and the persistence diagrams are computed up to dimension 3. Clearly, we can see that the clustering is not so strong in any one class as a whole because all the songs that we have are different from each other. But we can see that some songs within the class and when compared to the other classes are similar to each other, e.g. in dimension 1, artist 2 and artist 0 have similar homology

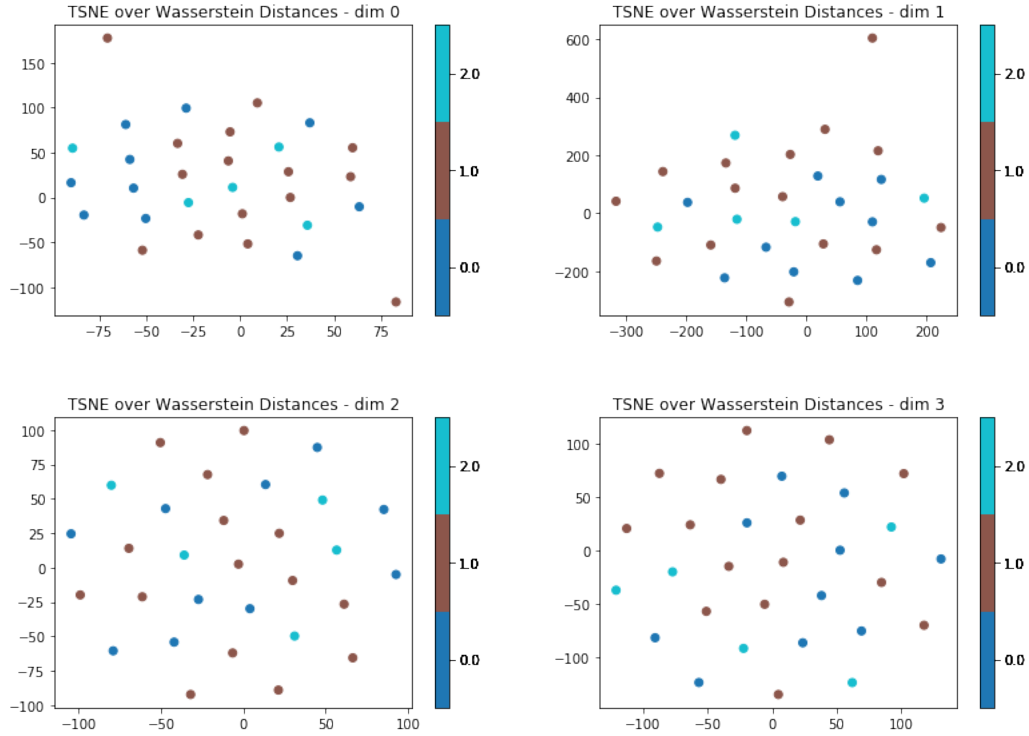

Figure 3: TSNE applied to Wasserstein distances over persistence diagrams. Here also, we can see that the clustering is not so strong in any one class as a whole because all the songs that we have are different from each other. Some songs are completely different than others, for eg. in dimension 0, two songs of artist 1 are very different from all other songs. But we can see that some songs within the class and when compared to the other classes are similar to each other, e.g. in dimension 2, artist 1 and artist 0 have similar homology

5 Updated Milestones, Any modification, and Project on Track?

At this point, we are not adding or removing any of our milestone. We believe that we are on track to complete our project by the end of the month. During these initial three weeks, we mostly spent our time learning about the structures in the music data as well as collecting and processing the data. We also tried a few basic tests to see if the data is reasonable or not.

6 Summary

Summary of our project is exploring and analyzing interesting topological structures and answer several real-world questions for the music data. Some of the questions we would like to address are the relation between chord progressions, genre classification, artist classification etc. We are using a diverse dataset so as to make it as close to the real world as possible.