AUDIO AND MUSIC PROCESSING LAB SESSION 8-9:

Duplicate Analysis

Instructor:
Alastair Porter

Student:
Erim Yurci

Instructor:
Dmitry Bogdanov

University Pompeu Fabra, Music Technology Group (MTG)
erimyurci@gmail.com

ABSTRACT

This paper is a report for the 8th-9th session of the class Audio and Music Processing Lab 2015/2016 in Sound & Music Computing Program, UPF. The aim of this work is to investigate large dataset of AcousticBrainz's duplicate submissions. In this task **bpm** values of the submissions are targeted. For each song a small dataset is created; Including every duplicate of that particular song. And the first two most used bpm values are plotted with their percentage over total number of duplicates. The results output as .CSV file for visualization and further analysis.

1. MATERIALS

For this assignment large "ab-duplicates1000-2016-03-02" dataset is used. Which consists of AcousticBrainz descriptors of 1000 songs and their duplicates. The dataset is provided as. .JSON files. The number of total duplicates processed is 42,991.

The programming language used in this work is **python**. The project code and resulting CSV file is submitted to the github. And could be found at: https://github.com/yurci/acousticbrainz_duplicate_bpm .

2. METHODS

Firstly, under the "duplicate_bpm.py" code duplicate submissions are traversed inside the "file_trav(inputDir)" function which takes "ab-duplicates1000-2016-03-02" folder as an input. From the file names unique MBIDs are determined (first 36 characters). And duplicates from the same song stored in a distinct array. That array send to the "json_Read(path_Array)" function for further analysis.

Secondly, inside the "json_Read" function, which takes an array containing paths of duplicates belonging to the same MBID, BPM values are read. These values stored in a 2-d array as 1st dimension representing the BPM value and 2nd dimension is the count. If a duplicate has same BPM value with any previously processed duplicate, only the count value is rose for the relevant BPM value.

Thirdly, two of the most used BPM values are detected with their counts and percentages.

Fourthly, a global check is applied in order to detect if the two of the most used BPMs are close to each other. I checked if BPMs are close to each other under +2, -2 range. If so; Second most used BPM is merged with the first and second most used BPM value and percentage marked as "Nan".

Finally, the obtained data from a unique song appended in a global array in order to output as .CSV file. Final output consists of these columns respectively: MBID, #1 Most Used Bpm, Percentage of 1st, #2 Most Used Bpm, Percentage of 2nd, Total # Duplicates.

3. RESULTS

49.6% of the songs consists of duplicates which has 100% same or within +2, -2 threshold same BPM value. **27.0%** of the songs duplicates has same BPM value between 99%-90% rate. Only **3.4%** of the submissions are below %50 rate and showed variety on BPM values.

4. CONCLUSIONS AND FUTURE WORK

We can say that most of the songs duplicates has same BPM value. Only **23.4%** of the songs consists of duplicates with different BPM value below 90% rate.

For the future "duplicates with close BPM value detection system" could be changed to be smarter. Also calculated Key values, in same manner, could be merged with BPMs. And we can detect duplicate submissions which are not actually duplicates but a different song.