520.445

Audio Signal Processing

Fall 2015

PROJECT 3

(Assigned 11/12/15, Due 12/11/15 at 11:59pm)

Project goals

In this project, you will build a song identification system, similar to that behind the Shazam iPhone App (to check it out, download it or ask a friend). Your task will be to construct a database of song fingerprints for a collection of songs. The fingerprints should be small memory footprint audio summaries that are easily searchable and that span the duration of each song. In conjunction with the fingerprint database, you will define a search algorithm will take as input a 3 second clip extracted at random from one of the songs and return the identity of the song the clip came from. This project will be somewhat open ended, with the nature of the solution left largely up to you. There are four interrelated considerations involved in the design: accuracy of the system, size of the fingerprint database (smaller is better), speed of the search algorithm (faster is better), and robustness of the identification (noisy search clips should work).

Files provided:

The mat-file musicDB.mat contains a 150-dimensional structure array named musicDB. Each element is a structure corresponding to a particular song containing the following fields:

1. title: a string with the songs title
2. genre: a string with the songs musical genre
3. signal: mono waveform containing the first 10 seconds of the song sampled at 16 kHz

Functions you must design

1. Filename: compute_fingerprints.m

Prototype: [fingerprints] = compute_fingerprints(musicDB) This function takes the data structure array as input, computes the fingerprint for each song, and returns a 150-

dimensional structure array containing fingerprints for the 150 songs (the contents of this structure array will depend on you).

2. Filename: identify_song.m

Prototype: [song id] = identify_song(clip,fingerprints) This function takes a three second clip (as a 480001 array) and the structure array of song fingerprint computed above. The contents of fingerprints structure will depend on you. The function should return the index of the song (i.e. a number from 1 to 150).

Report

Include all code, graphics, a precise formal description of your fingerprint design and search algorithm, and a detailed discussion of performance and the lessons learned from your effort. E-mail your report and code to the the course assistant Ashwin Bellur <abellur1@jhu.edu>. In descending importance, you will be graded on (1) the creativity and depth of your effort, (2) the quality and thoroughness of your report, and (3) the accuracy, memory footprint, and speed of your search algorithm. Keep in mind that your accuracy will be evaluated on a random selection of 3 second clips extracted from the signal fields that will NOT be provided ahead of time. Make sure that performance details (accuracy, memory, speed) are well detailed in your report.

A few pointers ...

1. Make sure you use the exact m-filenames and function prototypes specified above as they will facilitate automatic grading. Any deviation will result in a loss of points.

2. As this is a fairly open ended project, don't be discouraged if your system does not work as well as you'd like. What's most important is that you put in a dedicated effort and describe it thoroughly. The same goes for the bonus problems below.

3. It is important that you make a dedicated effort to design a fingerprint that is efficient both in memory usage and search speed. A trivial solution to this problem exists in the form of a fingerprint that is just the original waveform such a trivial solution or anything close to it is NOT acceptable. In other words, the size of fingerprints should be MUCH SMALLER than data.

Bonus points:

Noise robustness (30 points):

Design your fingerprint to be robust to additive white noise to as low an signal to noise ratio (SNR) as possible. Note that you should not artificially add white noise to the data before fingerprint construction, but instead design a fingerprint that will be inherently robust to an unknown noise level, while still maintaining good performance in clean conditions (i.e. if your identifier works well in 0 dB white noise, but crummy in clean, that is BAD). This portion will be graded on the quality of your effort and writeup and the performance of your identification using random clips with various levels of white noise added.

Genre Classification (40 points):

Design a feature set and a classifier that can be used to discriminate between pairs of songs of different genres (e.g. a rock vs. folk classifier). Consideration should be made to (1) which acoustic measurements/features are indicative of various styles of music and (2) which machine learning algorithm would be required to separate those examples in that space. For each distinction, use the first half of the examples of each class to train and the second half to test. This portion will be graded on the quality of your description and the performance of your classifier (performance must be well documented in your report).