CSS 584 Multimedia Database Systems
Autumn 2023

Assignment 4, 100 possible points (15%)

Peer preliminary demo/Q&A: November 27th (Monday) in class
Program & Report Due by: December 12th (Tuesday) 11:59 pm

Description:

This project has 2 options. You may pick any one of them based on your interest and strength. Option 1 can be done in groups of 2-4 students but Option 2 has to be done individually.

I. Option 1 is to implement an audio classification system that involves audio signal processing and machine learning. It may require some research and system integration (with existing open source machine learning programs). You will be given some audio files, research paper references and links of useful software. But you have freedom to develop your own approach and use other audio files. Self-signup on Canvas

Requirements:

1. Audio files
   You will need a group of speech files and a group of music files. You can also use the 40 short audio files (20 speech, 20 music) I uploaded to Canvas (inside Files/assignments/assignment4_supportFiles)
2. Operational Steps
   2.1. Develop program to read each audio file and extract its audio features. For a long audio clip (i.e., more than several minutes long), you are suggested to partition it into several segments and extract features for each segment. **For short audio clip** (as the ones in Canvas), you may treat each one as a single unit. You may read reference papers and try various features but a good start is the features we discussed in the class. You may use **NAudio** (https://github.com/naudio/NAudio) to sample the audio files for their amplitude, **Exocortex.DSP** (http://www.codeproject.com/Articles/2198/Exocortex-DSP), **Accord.NET** (http://accord-framework.net/docs/Index.html) or **Aforge.NET** (http://www.aforgenet.com/framework/) to perform Fast Fourier Transformations on samples to derive their frequencies. At least 3 audio features (among them, at least one from the frequency domain) should be extracted.
   2.2. To prepare for machine learning process, the data may be formatted as below:
   #, f1, f2, …, fn, label
   where # lists the file name, f1, …, fn are n features extracted from this file, and label has value "yes" if this file is a music clip and "no" if it is a speech file.
   2.3. Use about 2/3rd of the data (half representing speech files, half music) as training data to build music model. You program should be able to call machine learning algorithm to build a model (or models) on this data. There are some existing open source machine learning programs, such as SVM light (an implementation of Support Vector Machines in C, downloadable at http://svmlight.joachims.org/)

and WEKA (implementation of a collection of machine learning algorithms in Java, downloadable at http://www.cs.waikato.ac.nz/ml/weka/).

2.4. The rest 1/3rd files are used as testing data. Your program should be able to call the machine learning algorithm to apply the model on this testing data and show the following information: **#, Model output, Ground truth label**
where # is the file name, model output is the "yes" or "no" label given by the model, and group truth label is the correct label you provide in step 2.2.
You may print other additional information if you like.

2.5. On GUI, users can choose to listen any audio file in the project dataset so the program should support audio play function.

Preliminary demo/Q&A

1. Show system interface
2. Show audio files can be read by the program
3. Anything else you implement
4. May share and discuss libraries/tools/techniques you found useful, and have Q&A to help each other

What to be included in your report:

1. How to run the code, how to use the system, functionalities of each program file
2. List and briefly introduce libraries/tools/techniques you used in your development
3. List features and their values for the audio files and show what files are used as training data, what are testing.
4. Show comparison between model output and ground truth label (as discussed in 2.4) and display the precision and recall value of your model on the testing data.

NOTE: as this is a research oriented, open-ended project, no intermediate data or final results will be provided.

Evaluation Criteria: - 100 points

System design and integration:	20 pts
Correctness (testing):	50 pts
Report and comments:	30 pts

II. Option 2 is to implement a Video Shot Boundary Detection System. The algorithm will be discussed on Monday Feb. 27. It is listed below for your reference. You have to follow it exactly and test on the video provided.

Requirements:

The algorithm used is called Twin-comparison based approach. The main idea is discussed in Section 4. You may find a copy of the research paper on Canvas as reference.

1. Test Video

   Multiple versions of a same video 20020924_juve_dk_02a are provided, in .avi files with different codec, and in .mpg format (inside Files/assignments/assignment4_supportFiles on Canvas). They are provided in case you don't have correct codec in your computer or the program language you choose has better support on one format over the other.

2. Operational Steps

   2.1. Read video file. Note you program should be able to read the video sequence directly (i.e., cannot use tool to extract frames and your program then reads the pictures one by one)

   2.2. **For each frame (from frames #1,000 to #4,999. Make sure the frame number your program captured matches with the one shown in VirtualDub** – you may download it from Files/assignments/assignment4_supportFiles tools folder on Canvas), get histogram value (using #of pixels, not ratio) for 25 bins according to Intensity Method listed in Project1.

   Note: suggest to save this feature matrix H

   2.3. Get frame-to-frame different

$$SD_i = \sum_{j=1}^{25} \left| H_i(j) - H_{i+1}(j) \right| \quad (1)$$

   2.4. Set thresholds

   2.4.1. For Cut:

$$Tb = mean(SD) + std(SD) * 11 \quad (2)$$

   2.4.2. For Gradual Transition

$$Ts = mean(SD) * 2 \quad (3)$$

$$Tor = 2 \quad (4)$$

   2.5. Loop through SD

   2.5.1. If $SD_i >= Tb$, then cut starts at i (i.e. $Cs = i$) and ends at $i+1$ (i.e., $Ce = i+1$)

   2.5.2. If $Ts <= SD_i < Tb$, consider it as the potential start (Fs_candi) of a gradual transition. Continue to check its following SD values. The end frame (Fe_candi) of the transition is detected when its next 2 (i.e. defined by Tor) consecutive SD values are lower than Ts or reaches a cut boundary.

   2.5.3. If $\sum_{Fs\_candi}^{Fe\_candi} SD_i >= Tb$, then Fs_candi and Fe_candi are taken as real start (Fs) and ending (Fe) of a gradual transition. Otherwise, this section is dropped and the search continues for other gradual transitions.

2.6. **Outputs** the sets of (Cs, Ce) and (Fs, Fe).

3. System GUI
   3.1. Show first frame of each shot: considering Ce, $Fs+1$ the first frame of each shot (Cs, Fs the end frame of its previous shot). [Can reuse GUI you've developed]
   3.2. Allow user to view the corresponding video shot (i.e., the shot can be played) by clicking on its first frame.
4. Idea of Twin-Comparison Approach

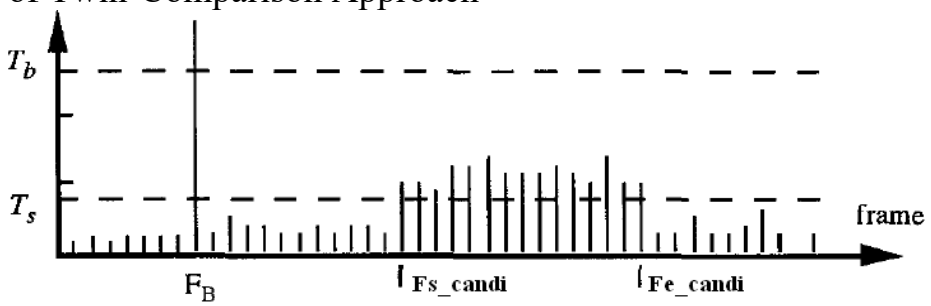

Figure 1. Illustration of twin-comparison

Twin-comparison requires the use of two cutoff thresholds: Tb is used for camera break (cut) detection. In addition, a second, lower, threshold Ts is used for gradual transition detection. The detection process begins by comparing consecutive frames using Eq. 1 to get SD. Whenever the difference value reaches threshold Tb, a camera break is declared, e.g. Fb in Fig. 1. The twin-comparison also detects differences that are smaller than Tb but greater than or equal to Ts. Any frame that exhibits such a difference value is marked as the potential start (Fs_candi) of a gradual transition (labeled in Fig. 1). The end frame (Fe_candi) of the transition is detected when its next continuous Tor (a preset threshold) numbers of SD values are lower than Ts or it reaches cut Tb threshold. Then if all the SD values between Fs_candi and Fe_candi add up to be greater than or equal to Tb, this duration is considered to contain a gradual transition. Otherwise this section is dropped and the search continues for other gradual transitions.

Preliminary demo/Q&A

1. Show system interface
2. Show video file can be read by the program
3. Anything else you implement
4. May share and discuss libraries/tools/techniques you found useful, and have Q&A to help each other

What to be included in your report:

1. How to run the code, how to use the system, functionalities of each program file
2. List and briefly introduce libraries/tools/techniques you used in your development
3. List the first frame # of each shot: considering Ce, $Fs+1$ the first frame of each shot (Cs, Fs the end frame of its previous shot). [Can reuse GUI you've developed]

Evaluation Criteria: - 100 points

Correctness (testing):	70 pts
Report and comments:	30 pts

III. Submission for both options:

The **softcopy of well-commented code (& an executable ready for testing to avoid the delay of submitting your final grade) and report** (See **What to be included in your report** Section inside each option) needed to be submitted to Canvas by the due time. You will also be asked to do a peer preliminary demo (see sections **Preliminary demo/Q&A** inside each option).