1 Project Task 2: Near duplicate detection

Create a spatio-temporal feature within a spatio-temporal cuboid

In this task you will provide as input a small video clip. Your algorithm will search your video database and find all the videos containing the same clip. It should also report the interval at which the same clip was found. The algorithm should be robust to size, contrast, aspect-ratio and other such degradations in the video. It should be noted here that unlike previous task we do not require similar videos, instead we aim to find the duplicate videos. For e.g. a few minutes intro of a Big Bang Theory is repeated in each of its episodes. This means that if I provide as input the few minutes intro of the said show, I should be able to get all the episodes of that show. Similarly, talk shows like Aapas Ki Baat or The Daily Show etc all have same set being captured from similar camera location and parameters in each of its episodes. Given one particular shot of the should result in retrieving all the episodes of the show and should also indicate on which set of frames the input shot and its duplicate was found.

Similar to the previous task, here as well you are required to process video shot by shot. video in your database as well as your input video could consists of multiple shots. Your algorithm should match all the shots of the input video with the videos in the dataset and establish whether the entire input video or portion of it is was found in the database. Your algorithm is thus required to match videos shot by shot.

1.1 Base Line Approach

1.1.1 Training

1. Go to each video in your database
2. Segment video into multiple shot
3. Project each shot in a feature space
4. Store the shot information and their feature representation in a file

1.1.2 Testing

1. Provide as input a single shot from a video
2. Project this shot into a feature space that you used for training
3. Go to each video in database and perform comparison
4. For each of the matched shot store a representative image of the found shot as well as its start and end frame location and video name on which it was found for evaluation.

1.2 Final Approach

Your final algorithm should be able to accept videos with multiple shots i.e. it should be able to perform near-duplicate detection between multi-shot videos. The training part may will remain the same however your testing algo could be modified as follows:

1.2.1 Testing

5. Provide as input a video
6. Segment the input video into multiple shots
7. Project each shot into a feature space that you used for training
8. Go to each video in database and perform comparison between multiple consecutive shots of the videos
9. For each of the matched video store a representative image of the found shots as well as its start and end frame location, portion of the video matching (for e.g. n out of m shots matched) and video name on which it was found for evaluation.