Final Project

Author:
Seyed Kamyar Seyed Ghasemipour

Dec. 7, 2015
1 Introduction

The project that I chose was Project 1 from the suggested projects provided. The main components of this project were:

- Shot Detection
- News Company Logo Detection
- Face Detection
- Face Tracking
- Male/Female Classification

Each of the sections below are divided into three sections:

1. Problem Description
2. My Implementation
3. Analysis of Results
4. Ideas for Improvement

2 Shot Detection

2.1 Problem Description

The task of shot detection is determining when there is a discontinuity in the camera motion. For example, in a scene where two people are speaking, the camera may show the face of one person in one frame, and show the face of the other person in the next frame; these two frames determine a shot boundary and would be considered to be of two different shots.

2.2 Implementation and Analysis

For this task, I used histogram based methods to determine the shot boundaries. Initially, the algorithm I used was as follows. Given two consecutive frames, two integers \( m \) and \( n \), and a distance measure \( \chi \):
1. Divide the frames into $m \times n$ components by subdividing them vertically into $m$ uniform sections and horizontally into $n$ uniform sections.

2. Calculate the histogram of the grayscale versions of the components of each frame.

3. Calculate the distance between the two histograms of corresponding components in the two frames.

4. For each pair of corresponding components, if the distance of their histograms is above threshold $t$, count a vote for shot change.

5. If the number of votes is larger than 50%, mark the frames a determining a shot boundary.

This algorithm was implemented by using the $L_1$ and $\chi^2$ distance metrics. Below I have included a screenshot of the execution of the algorithm. In this instance, the number of components was chosen to be $3 \times 4$ and the threshold was set at 5.
In the top left corner we can see a grayscale frame from the second clip provided and in the bottom left, its corresponding histogram. The screenshot also shows the histogram distances in consecutive frames, over the entire image, throughout clip 2 using $L_1$ and $\chi^2$ distance metrics. We can also see the for the components as well as the total number of votes.

Experimenting with different values for $m$, $n$, threshold as well as different distance metrics, I decided to use the $L_1$ distance over the entire image with a threshold of 400 (which is about 0.1% of the number of pixels). These hyperparameters were chosen by empirically looking at clips 1 and 2.

- $\chi^2$ distance over the entire image is very robust to small changes, in the image. However, it also overlooks some shot boundaries.
- Using multiple components instead of the entire image has shown to be too sensitive as can be seen from the bottom right plot in figure 2.
- The absolute value over the entire image seems to be a good middle ground between sensitivity and robustness.
Another aspect that I took into consideration was transitions between shots. Transitions cause spikes in the distance measures for a duration of a few frames. Since transitions cannot really be considered to be shots, we consider any significant increase in the measure that lasts more than one frame to be a transition.
2.3 Results

Putting everything together, I compute the F1 score for shot detection as well as well as transition detection. The table below shows the scores in each clip.

<table>
<thead>
<tr>
<th></th>
<th>clip 1</th>
<th></th>
<th>clip 2</th>
<th></th>
<th>clip 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prec.</td>
<td>Recall</td>
<td>F1</td>
<td>Prec.</td>
<td>Recall</td>
</tr>
<tr>
<td>Shots</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.50</td>
<td>0.67</td>
</tr>
<tr>
<td>Trans.</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.75</td>
<td>0.43</td>
</tr>
<tr>
<td></td>
<td>Prec.</td>
<td>Recall</td>
<td>F1</td>
<td>Prec.</td>
<td>Recall</td>
</tr>
<tr>
<td>Shots</td>
<td>0.4</td>
<td>1</td>
<td>0.57</td>
<td>0.56</td>
<td>0.83</td>
</tr>
</tbody>
</table>

Table 1: Precision, Recall, and F1 scores for each clip (L1 distance)

Combining multiple distance measures should improve our result; maybe learning a weighted combination of them on some training data? Also, In general, higher frame rates would most likely make this approach work quite well (for shot detection at least).

2.4 Ideas for Improvement

The method that I used here pays no attention to the semantics of the image, and also does not consider what the focus of attention in the frame is. Therefore, there are many situations in which it cannot perform well and incorporating semantic information should give a significant performance boost.

Below are two examples of situations where semantic information can be very helpful:
2.4.1

Consider the sequence of frames below were a Marvel logo passes by to indicate a transition. This situation is almost identical to if we had a shot of a motorcycle from the side going down the highway and a for a second a truck comes in between the camera and the rider. Being able to understand what is happening would help resolve this issue.
2.4.2

Another bad situation arises in clip 3. In the two sequences below, there is something held constant in the middle of the frames, but the rest of the frame is changing. However, we do not consider the one on the left to be a shot change whereas for the one on the right we do. Again, understanding what is happening is necessary for making this distinction.
3 News Company Logo Detection

3.1 Problem Description

Given a frame, we would like to detect the logo of the news company in that frame.

3.2 Implementation

I implemented this by filtering the image across multiple scales, using both the NCC and SSD measures.

3.3 Results and Analysis

This method provided no interesting results. Therefore, I have not included the results here.

3.4 Ideas for Improvement

Due to time constraints, I was not able to implement other ideas I had for this problem. Some of them are mentioned below:

- Using SIFT to match the images of the logos to some portion of the frame. However, there are two reasons why this may not work. First, logos generally have very flat color, so we might not get many SIFT keypoints in the frames. Second, even though SIFT keypoints are meant to be scale invariant, news logos are generally very small, which might further reduce the number of keypoints we get in the frame.

- Another method I was considering testing out was to take the logo of the company and embed it in random images. This would make up the positive set of images. Next we would take other random images as the negative images. Once the data is gathered, we can train a classifier and run it over the images using a sliding window approach over various scales.
4 Face Detection

4.1 Problem Description

The problem of Face detection is to find faces in images and mark them using a bounding box. Many methods exist for approaching the problem such as using DPMs, template matching, or using neural networks.

4.2 Implementation

For face detection, I used code from [http://www.ics.uci.edu/~xzhu/face/](http://www.ics.uci.edu/~xzhu/face/). This code implements the method described in *Face Detection, Pose Estimation and Landmark Localization in the Wild* [1] by Zhu et al. Below is a short description of their algorithm:

- Before anything, they pre-define the landmarks that they will be looking for.
- A HoG descriptor is learned for each of these landmarks.
- They consider the face to be a tree whose vertices are a subset of the landmarks and the edges connect them in a manner that forms a tree. In other words, each face model is a deformable parts model whose parts are a subset of their landmark vocabulary.
- They have one such model of a face for several orientations that a face might appear in.
- From data, they learn their model in a supervised fashion, optimizing a special function that define for their purpose.

4.3 Results and Analysis

This model works very well in the experiments I ran. However, the algorithm is very slow. Running on CPU, faces take about 30 seconds to be detected per image. For this reason, I was not able to tune the sensitivity of the model, and used the default settings. I found the default settings to have perfect precision, but there were frames in which recall was not perfect. This could also
be due to the fact that the faces sometime were too small. The documentation of the code states that it works best for faces larger than 80-by-80 pixels.

The algorithm returns the facial landmarks of a detected person. To get the bounding box, I calculate the centers of each landmark, then take the maximum and minimum of the centers in the $x$ and $y$ axes. Below are some sample detections:

![Figure 4: Successful detections](image1)

![Figure 5: Model failed to detect the two people in the middle](image2)

5 Face Tracking

5.1 Problem Description

Given the bounding boxes of the faces we get from the detection model, we would like to track the face of the same person throughout that shot.
5.2 Implementation

For tracking, I used the dynamic programming method from A5. The algorithm was adjusted to run only on frames with detections in them, and to run within shot boundaries.

**mini-rant: I spent approximately 12 hours altogether to debug the code and make it work within this context. I’m not sure why, but I felt like I had to mention that. :P**

5.3 Results and Analysis

The results of the tracks are quite decent. The results can be found in here https://www.dropbox.com/sh/wk6krelmr6l3tz2/AABIA-Fre1ht3NJk80djjz9Ea?dl=0. Improving the quality of the shot detection will improve the results of this section as well, since the tracks are constrained to stay within shots. Additional improvements could be made by taking care of situations when there is a detection miss for one frame, and the same person is considered as two separate tracks.

6 Male/Female Classification

6.1 Problem Description

Given the tracks we find in the clips, we would like to classify the person being tracked as male vs. female.

6.2 Implementation

We were provided with pictures of males and females with information provided regarding some facial landmarks. Instead of using them to find the bounding boxes of the faces, I ran the algorithm for face detection from part 4 on these images as well, and computed the bounding boxes in the same manner as was mentioned in 4.3. In this way, the bounding boxes of the training images are derived using the same method as the frames in hopes that it would help classification performance.

The male faces were randomly split into train and validation sets with 198 images used for training, and 58 images for validation (2 images did not
elicit a detection). The female faces were randomly split as well, with 200 images used for training, and 60 used for validation.

The models I trained were SVM and $k$-NN. The performance of SVM with an RBF kernel was 82.2% on the validation set. Below I have also included a plot of the performance of $k$-NN for various $k$s:

![Figure 6: performance of $k$-NN for different values of $k$](image)

Also, performance of SVM with boxes of size $100 \times 100$ was almost identical to performance with images of size $32 \times 32$.

I chose SVM as the model to use. Then, given a track, I ran SVM on the bounding boxes at every frame. Then the class with majority vote was chosen as the class of that track.

## 6.3 Results and Analysis

The results for this implementation can be found here [https://www.dropbox.com/sh/wk6krelmr6l3tz2/AABIA-Fre1ht3NJk80djjz9Ea?dl=0](https://www.dropbox.com/sh/wk6krelmr6l3tz2/AABIA-Fre1ht3NJk80djjz9Ea?dl=0).

If we had more data available, we could try and train a neural network model. I believe this would result in a good performance boost.
References