Topics in Unsupervised Learning: Video Object Segmentation

Rotem Ezra, Neta Elmaliach

20/06/2023

1 Introduction

Video object segmentation is a computer vision task that involves separating and extracting specific objects of interest from a video sequence. In video object segmentation, the goal is to assign a binary mask to each pixel in the video, indicating whether it belongs to the foreground object or the background.

The purpose of this project is to develop an object identification algorithm using the DPMMClustersStreaming algorithm. The algorithm aims to identify a specific object in a video by analyzing its characteristics and distinguishing it from the background. The objective of the algorithm is to accurately identify and highlight the object of interest in different frames of the video. The algorithm accomplishes this by utilizing the DPMMClustersStreaming algorithm, which clusters pixels based on their similarity and assigns them to different groups.

The dpmmpythonStreaming library focuses on semi-supervised video object segmentation, where the initial frame contains a mask of the object, and subsequent frames are segmented in an unsupervised manner. The DPMMClustersStreaming algorithm can benifit our project since each cluster is associated with a set of Gaussian distributions that represent the visual characteristics of the pixels within that cluster. These Gaussian distributions are used to make decisions about where a pixel should be assigned during the segmentation process.

2 Implementation

Processing the Video

Using the open-cv libary in Python we were able to open the video and extract the first frame. We displayed the first frame and let the user select the object by using the selectROI algorithm (from open-cv).

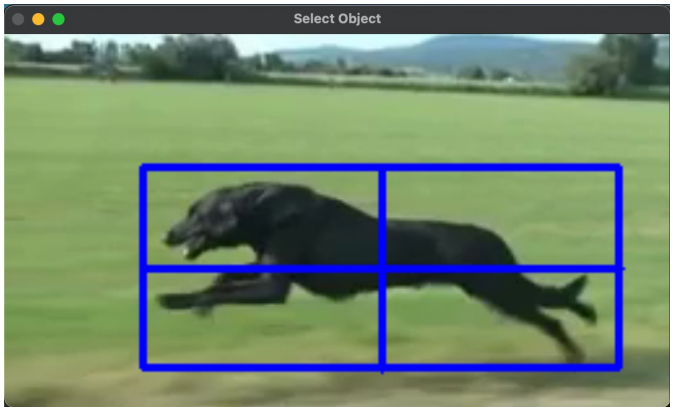

Figure 1: Selecting the object from the first frame

Afterwards we created a binary mask where the foreground pixels (object) are 1 and the background pixels are 0, by using the grabCut algorithm (from open-cv).

Figure 2: The object without the background

Figure 3: The background without the object

Preprocessing the First Frame

Now we have two images - one is the object in the first frame and one is the background in the first frame. These images are stored in a RGB 3D array and to fit the functions we want to use, we need to change them to a 2D array. To do this, we wrote a helper function called 'pre process image'. If the original image dimensions were (x,y,3), this function fits the data into an array of dimensions (5, x*y).

Creating the Model Using dpmmpythonStreaming

We created two models, one for the object and one for the background using the function 'fit init' from the dpmmpythonStreaming libary.

```
# on the first frame we need fit_init
obj_model = DPMMPython.fit_init(obj_data, 100.0, prior=prior, burnout=5, gt=None, epsilon=0.0000001)
bgd_model = DPMMPython.fit_init(bgd_data, 100.0, prior=prior, burnout=5, gt=None, epsilon=0.0000001)
```

Figure 4: Creating the models

The object data and the background data are the reshaped images. By using each model we can extract the clusters, and therefore the mean and variance of each cluster. This information will be useful for analyzing future frames.

Analyzing Each Frame

On every entry to the loop, we read the next frame to be analyzed. For each frame, we go over each pixel and determine if it is part of the object or part of the background. We do this by using a helper function 'is pixel of obj'. If the function returned that the pixel is part of the object, we will add the pixel data to an array that stores the updated object data. We will also add color in the same place for highlighting the object. If the function returned that the pixel is part of the background, we will add the pixel data to an array that stores the updated background data. A new frame is created with added highlights on each pixel that the function decided was part of the object, and we will add this new frame to the output video. This will result an output video of the original video with the highlighted object.

Analyzing Each Pixel

The helper function uses the function 'multivariate normal pdf' from scipy.stats libary. This function is is a mathematical function that allows us to calculate the likelihood of observing a particular set of values in multiple variables that are normally distributed. Our helper function receives as input the array of means and variances of each cluster from the first frame. Using a loop, the function calculates the probability of the pixel being part of the object for each object model and the probability of the pixel being part of the background for each background model. The function compares the maximum likelihood estimates for the object and background. If the maximum likelihood estimate for the object is higher than that of the background, the function returns True, indicating that the pixel is likely part of the object. Otherwise, it returns False, indicating that the pixel is likely part of the background.

3 Summary

The objective of this project was to develop an object identification algorithm using the DPMMClustersStreaming approach. The algorithm aimed to accurately identify a specific object in a video by using clustering techniques. We used the dpmmpythonstreaming Python library to implement the algorithm, along with OpenCV for video processing tasks and scipy.stats for probability density calculations. To accomplish the task, we divided the video into frames and analyzed each frame. By analyzing each pixel in any frame we were able to identify the object of interest. Despite encountering some challenges, such as false positive detections in certain frames, the algorithm demonstrated sufficient performance overall.

Figure 5: Example frame from the output video

In conclusion, this project successfully applied the DPMMClustersStreaming algorithm for object identification in videos. The project had a good outcome, but it also revealed the need for further improvement to make it more accurate.