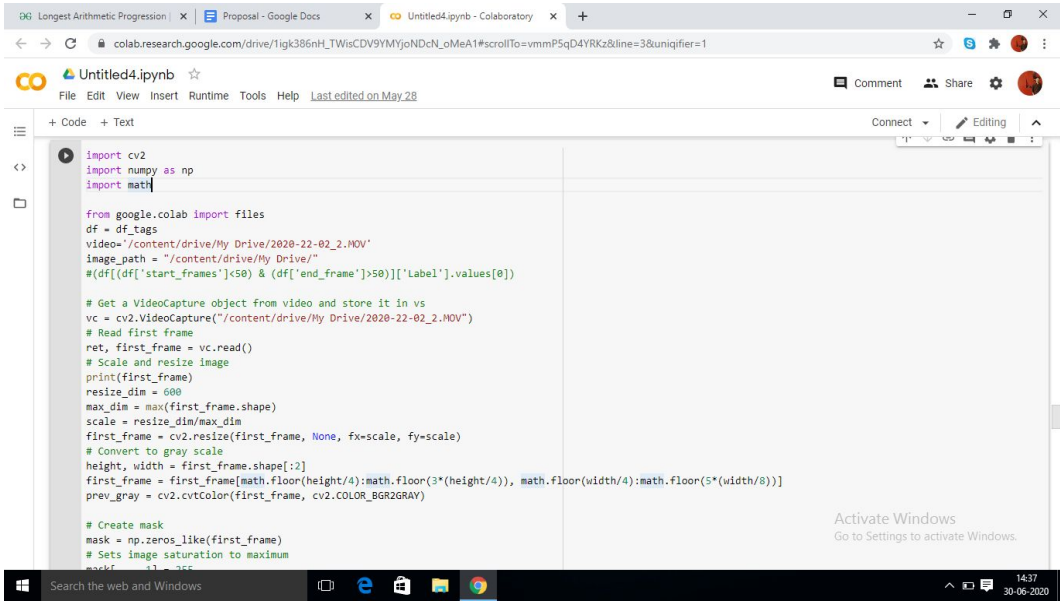
Problem Statement

Objective:

Crop out the clips from the provided videos where batsmen played a shot through supervised computer vision models.

Steps:

1. Calculate Optical Flow between consecutive frames(at a difference of ½ seconds).



Longest Arithmetic Progression | Proposal - Google Docs | Untitled4.ipynb - Colaboratory

colab.research.google.com/drive/1igk386nH_TWisCDV9YMYjoNDcN_oMeA1#scrollTo=vmmP5qD4YRKz&line=3&uniqifier=1

Untitled4.ipynb

File Edit View Insert Runtime Tools Help Last edited on May 28

Comment Share

+ Code + Text

Connect Editing

```
# Sets image saturation to maximum
mask[..., 1] = 255
# cv2.imshow('a',first_frame)
# cv2.waitKey(5000)
# cv2.destroyAllWindows()
video = video.split('/')[-1]
#print (video)
#out = cv2.VideoWriter('video.mp4',-1,1,(600, 600))
images = []
labels = []
success = True
fps = int(vc.get(cv2.CAP_PROP_FPS))
count = 1
prev_frame = count
curr_frame = count
folders = ["shots", "non_shots"]
clip_prefix = "clip_"
subfolder_fix = "/"
clip_ext = ".jpg"
loc = 0
#path = os.getcwd()
#for folder in folders:
#    if not os.path.exists(os.path.join(path, folder)):
#        os.makedirs(os.path.join(path, folder),exist_ok=True)
while(success):
    # Read a frame from video
    succes, frame = vc.read()
    if success == False:
        break
```

Activate Windows
Go to Settings to activate Windows.

Search the web and Windows 14:38 30-06-2020

Longest Arithmetic Progression | Proposal - Google Docs | Untitled4.ipynb - Colaboratory

colab.research.google.com/drive/1igk386nH_TWisCDV9YMYjoNDcN_oMeA1#scrollTo=vmmP5qD4YRKz&line=3&uniqifier=1

Untitled4.ipynb

File Edit View Insert Runtime Tools Help Last edited on May 28

Comment Share

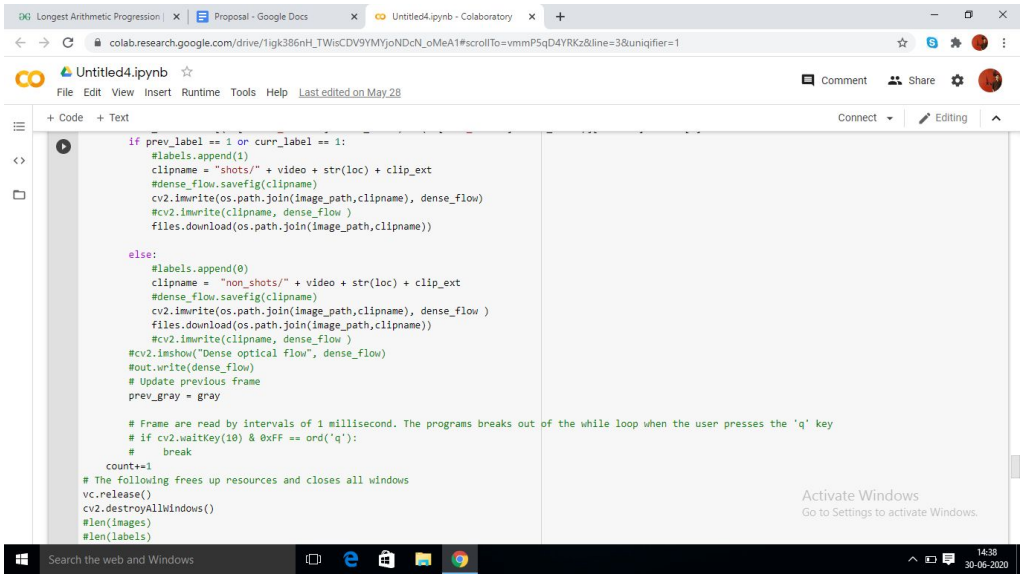
+ Code + Text

Connect Editing

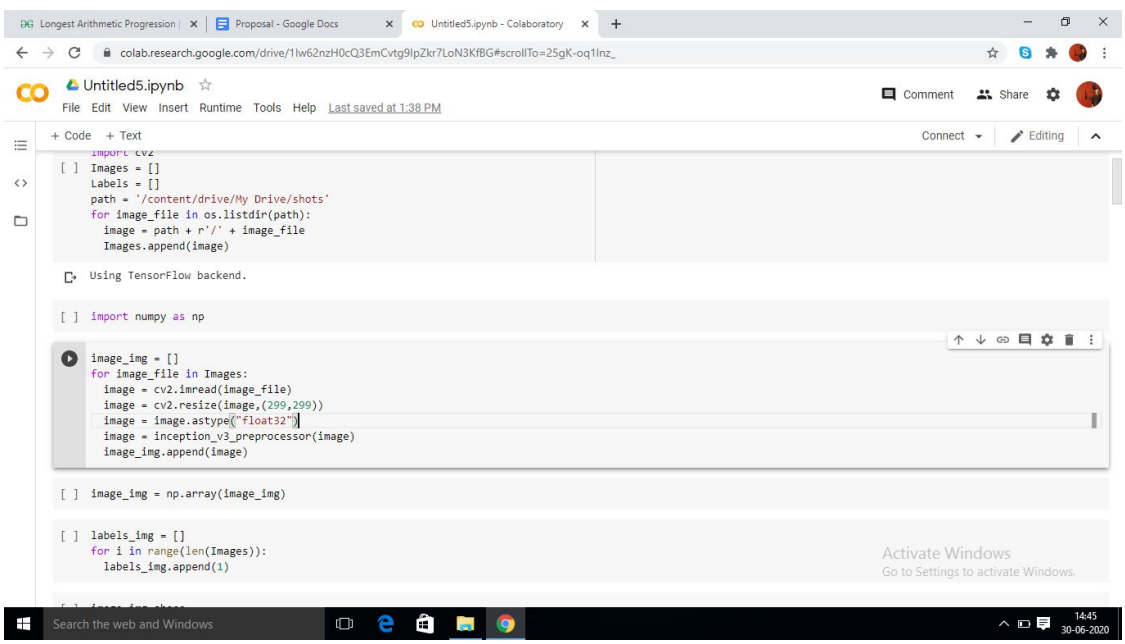
```
    elif count%(fps/2) == 0:
    # Convert new frame format`s to gray scale and resize gray frame obtained
        loc= loc+1
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        gray = cv2.resize(gray, None, fx=scale, fy=scale)
        gray = gray[math.floor(height/4):math.floor(3*(height/4)), math.floor(width/4):math.floor(5*(width/8))]
        #prev_gray = prev_gray[0:height, math.floor(width/4):math.floor(3*(width/4))]
        # Calculate dense optical flow by Farneback method
        # https://docs.opencv.org/3.0-beta/modules/video/doc/motion_analysis_and_object_tracking.html#calcopticalflowfarneback
        flow = cv2.calcOpticalFlowFarneback(prev_gray, gray, None, pyr_scale =0.5, levels = 5, winsize = 11, iterations = 5, poly_n = 5, poly_sigma = 1.1, flags
        # Compute the magnitude and angle of the 2D vectors
        magnitude, angle = cv2.cartToPolar(flow[..., 0], flow[..., 1])
        # Set image hue according to the optical flow direction
        mask[..., 0] = angle * 180 / np.pi
        # Set image value according to the optical flow magnitude (normalized)
        mask[..., 2] = cv2.normalize(magnitude, None, 0, 255, cv2.NORM_MINMAX)
        # Convert HSV to RGB (BGR) color representation
        rgb = cv2.cvtColor(mask, cv2.COLOR_HSV2BGR)

        # Resize frame size to match dimensions
        frame = cv2.resize(frame, None, fx=scale, fy=scale)
        frame =frame[math.floor(height/4):math.floor(3*(height/4)), math.floor(width/4):math.floor(5*(width/8))]
        # Open a new window and displays the output frame
        dense_flow = cv2.addWeighted(frame, 1,rgb, 2, 0)
        #images.append(dense_flow)
        prev_frame = curr_frame
        curr_frame = count
        prev_label = df[(df['start_frames']<=prev_frame) & (df['end_frame']>=prev_frame)]['Label'].values[0]
        curr_label = df[(df['start_frames']<=curr_frame) & (df['end_frame']>=curr_frame)]['Label'].values[0]
```

Activate Windows
Go to Settings to activate Windows.

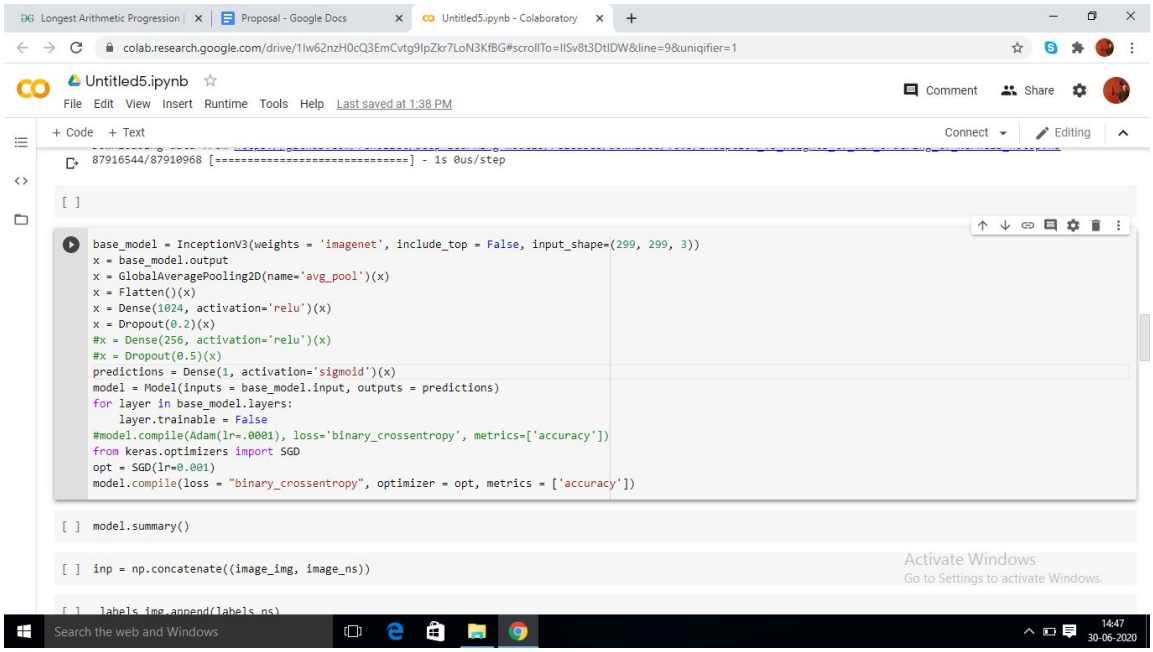
Search the web and Windows 14:38 30-06-2020



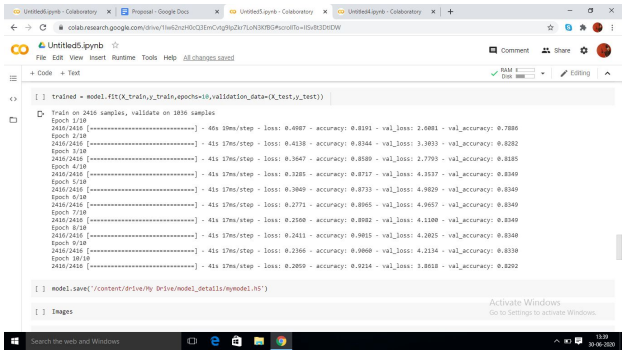
2. Apply Image preprocessing techniques on captured Optical Flows.



3. Apply transfer-learning from pretrained models (VGG16, Inception-v3, etc) to extract features.



4. Train a binary classifier DL model to predict inference from each Optical Flow Image - Whether the shot is being played or not (WIP Updates: classification 82% accuracy recorded).



Dataset(s)

Overview:

3-4 hours of videos during training sessions (equals approx. 640 cricket shots)

End to End Model

Input:

The video for which highlight is to be calculated.

Output:

The output of the model will be a list of labels(0- corresponding optical flow image is not a part of shot & 1- corresponding optical flow image is a part of shot). There will be a total of (video length(in seconds)*2) entries in the corresponding list of labels.