**Computer Vision Assessment**

Approach:

1. Preparation of the dataset.
2. Training and selection of the object detection model.
3. Real Time detection and post processing (i.e)…
   1. The trained weights of the object detection model must be taken.
   2. It should be applied on the raw video.
   3. Segregation of the Region of interest(ROI) (or) the bounding boxes area should be segregated.
   4. Then the OCR algorithm is applied on these ROI’s and the text is detected
   5. The recognised text is then appended in a DataFrame and that is converted to an excel file and downloaded as a file.
   6. This downloaded excel file is once again fed into the model to aggregate the all the text as a single word and all the time frames are combined (i.e) if there are 30 frames per second then for each frames many or few characters maybe detected, these are now joined as a single word, which means that all the frames are combined together for a particular second and so on for the whole video combining all the characters into a single word and it gives the final output file.

Challenges Encountered (Explanation):

1. Preparation of the dataset.
   1. This is most time consuming task I’d say, cuz the dataset was not easily preparable. I searched the web in the most common sites for the dataset containing the images of shop signs like *Open Images Dataset(google), COCO Dataset, Data world etc….* but those sites don’t contain datasets similar to this task.
   2. Thanks to Roboflow, they had few datasets which was specific to this problem, but the problem here was the annotations in some of the images were wrong so I downloaded the dataset from the website and did webscrapping to get some more similar to Indian malls and did some processing on the scrapped images before adding on the real dataset and started doing the annotations. This took me a long time and also frustrating while it got some lags in the website.
2. Training and selection of the object detection model.
   1. We can also train an object detection model there and the graphs they provide are very much interactive graphs etc… but we can’t download the weights of the trained model, so I had to do it in google collab.
   2. I always had a wish to train an object detection model from scratch so I followed a youtube tutorial from Krish Naik: *https://youtu.be/XoMiveY\_1Z4?si=2tfUaXrD4Jcj1NTI* . But it didn’t work for me. So I had to move on to YOLO model, cuz I already know the procedure on how to train as I’ve done an project on that.
   3. So I uploaded my dataset in my drive and started the training of the YOLO model with medium no of parameters “*YOLOV8m.pt”* but the collab crashed after 200 epochs. And I tried again from another account as the limits we over here and got the same result.
   4. So I moved on the smallest no of parameters model *“YOLOV8n.pt”* and during the first two runs the kernel crashed and during the third run after 300 epochs the model was fully trained and I was able to download the weights of the trained model.
   5. The training and waiting time was long during this but I got some to play with OCR techniques and got to know which gave the best results when applied post processing on to the frames of a video.
3. Real Time detection and post processing:
   1. I downloaded the this video  <https://www.youtube.com/watch?v=UKSR0XNSXSo> from youtube and trimmed it to 5 mins and did the post processing steps on it.(*I’ll provide the link to the video and the excel file below.*)
   2. As I mentioned earlier during the training of the YOLO model I went through the post processing steps for text recognition and noticed that pytesseract gave good results with just converting the given input image into gray scale images better than apply morphological transformation, hue etc…. testing it by taking a screen shot of an image from the video
   3. First my approach was to apply the trained YOLOV8 algorithm on the raw video, get the bounding and develop another model to detect the bounding boxes and the text inside it.
   4. But while I was going thorugh some OpenCV tutorial they provided the size of the bounding boxes to be predicted using pytesseract (OCR library). So I thought why not integrate the pytesseract library with the object detection model.
   5. I took the Region of Interest(ROI) or Bounding Boxes co-ordinates and applied the OCR on the ROI’s and ran the model, on the 5 mins video and that I uploaded in collab and read the video frame by frame using opencv and nearly after 1hr:30mins the kernel crashed, the output video which was being created was also lost cuz all the variables were erased.
   6. So I installed all the dependencies in my local pc and ran the code in jupyter notebook and after 3-4 hours of running I was able to get the video and excel file for that 5 mins video, 1mins video…..
   7. Now if the video was 30fps, then for each frame one or more characters maybe recognised and when we integrate them as a single frame (one second as whole) the characters get repeated like some word which does not make sense, only human eye interpretation could guess what word is it. But this is not a good thing our model should be robust enough do all these things right? So I took an online tool reduced the 30fps to 1fps and ran the model again to detect the texts and provide the excel file but now the detections accuracy was very low due no continuation (I think so, but the detection accuracy sure reduced.)
4. Solution and future steps:
   1. So I went to chatgpt to overcome this it said to tune a **LLM**, given a corpus a large text data it gives a text a similar to that, which means given a word with no meaningful representation it gives a similar word which is in the corpus. So I downloaded a text dataset from kaggle to do this but just gave it a thought, cuz how the LLM’s work is they embed the given corpus of words into a vector which is called as text embedding and this is done by calculating the cosine similarity between the given words. But in our problem even if we give a dataset of company names the names we provide for testing has no meaning cuz the model while testing also embeds the input words on to a embedding same and gives the output similar to that space where it is embedded, hence this don’t give us a good solution.
   2. So it gave another idea of **Named Entity Recognition (NER)** and I tried it and it didn’t work.
   3. So my solution for this is a brute force solution and I haven’t tried it yet cuz it takes more time to prepare the dataset for this model. The solution is train a CNN architecture or take a pretrained model like VGG-16(it gave good results for one of my tasks which I did), and take all the images of shops(signs or boards) and do various augmentations on the images of the particular shop(shop sign or board) and store it as a separate class, which means for each shop each class is created and the CNN architecture is trained on it and after taking these ROI’s the CNN model is applied on the ROI instead of the OCR module and it gives a class as output which will be full meaning or full word.