**HOW TO USE THE CODE :**

**Preprocessing of the dataset:**

1. First of all we need to resize the image to (160, 576) as this is the input shape of the pretrained VGG-16 networks. The file “resize\_red\_masking.py” will be helpful in this case.

2. After this we will manually create the mask of the image. In this procedure we will paint the road as red or set the pixels value of road to [0,0,255] (BGR format). This can be done using simple Paints software.

3. The below shown image is a sample of it. (Left side is image and right side is its mask).

After this is done, we will again use the file “resize\_red\_masking.py” to give the road part (red pixels) pink and and the whole background as red color. The reason behind this is, that the code work is largely inspired by the kitty dataset which has same format of segmentation.

After the above step, one will get output which will be similar to the below shown image. However, if there is any noise in the image, then one can remove it by manually painting it.



During this process, it will rearrange the directories of image files in ascending order and rename

the files to “umm\_road\_i.png” format (where, “i” is an integer). This helps to arrange the image and

its corresponding semanted version in same order.

**Training the model:**

To train the model, first make sure that your folder the below shown directory.

Here,

1. checkpoints folder -> This will contain the pretrained model.

2. data folder -> This will contain the dataset and pretrained weights of the encoder part (VGG-16 part). This will be stored as “vgg” folder under “data” folder.

You can download weights from following link : -->

https://s3-us-west-1.amazonaws.com/udacity-selfdrivingcar/vgg.zip

.

├── checkpoints

├── data

│   ├── data\_road

│   │   ├── testing

│   │   │   ├── calib

│   │   │   └── image

│   │   └── training

│   │   ├── calib

│   │   ├── sem\_image

│   │   └── image

│   └── vgg

│   └── variables

├── image\_predicions

├── \_\_pycache\_\_

└── video\_predictions

Under, the data folder we will have two folders one for training and othre for testing. The training folder will further have two folders one is “image” folder for the input image and “sem\_image” for the semanted images”. In testing folder we will have only one folder which is “image” and this will contain our testing image.

3. image\_predicion and video\_prediction folders -> These folders will simply contain your outputs from the model.

To train your model simply run the below command ->

python combo\_run.py

One can tune the epochs size and batch size in the “main\_runner” function of this file.

Once the training gets over, the model will get saved to checkpoints folder.

**Image prediction:**

1. To test the model on your images, put the images in the “testing” folder of “data\_road” folder (as stated earlier”).

2. After this turn the “training\_flag” to False, in the if \_\_name\_\_ == '\_\_main\_\_': (This is present in last lines of the same file.

3 The outputs will be stored in “image\_prediction” folders.

**Video prediction:**

1. To see the predictions of the model on the video. Open the “combo\_test.py” file. Set the arguments of the video feed in this file. (The VideoCapture of OpenCV module).

2. Once this is done, you can turn of the “SAVING “ flag to False if you want ot see only the see the segmentaion ans not want ot save it. Turn it to True if you also want to save your video predictions.

**Real Time prediction:**

1. For real-time segmentations, just set the VideoCapture argument to “0” (this zero signifies the camera which you want ot use for this process”

2. The rest work is same as the Video prdiction part.

**Requirements:**

1. Python == 3.7.11

2. Tensorflow == 2.0.0

3. Keras == 2.3.0

4. Scipy == 1.2.0

5. OpenCV == 4.5.5

Rest of the dependencies will get accordingly downloaded as per the python verison.

**Commands to run the code:**

**1. Setting up the virtual env:**

Run the following codes in your terminal to set up the virtual env.

**conda create -n myenv python=3.7.11**

**source ~/miniconda3/etc/profile.d/conda.sh**

**conda activate myenv**

**conda deactivate # deactivate env.**

**2. Preprocessing:**

python resize\_red\_masking.py

(but remember to comment the masking part, only resizing done, here).

**3. Manual Masking: Done via common Paints software by Windows (Paints 3D).**

**4. Preprocessing – 2:**

python resize\_red\_masking.py

**5. Training:**

python combo\_run.py

**6. Image Testing:**

python combo\_run.py

# In the above step, turn off the training\_flag command.

**7. Video Testing:**

python combo\_test.py

# Do this by adjusting the videos directories.

**8. Real Time Testing:**

python combo\_test.py

# Do this setting the VideoCapture argument to “0”.