Assignment 1

Sangeeta Yadav (15318)

February 25, 2022

All the source codes are available here.

1 Question 1

The NCut algorithm is implemented from scratch in Ques1 Ncut.ipynb. The trickiest part was the eigenvalue decomposition, so I have used np.linalg.eigh which exploits symmetry properties of the matrix. All the images are resized to (64, 64) so as to save memory as the weight matrix was quite large.

1.1 Implementation Details

Following feature representations are used for each pixel:

- HSV Features: $[v, v * s * sin(h), v * s * cos(h)]$
- RGB Features: $[r, g, b]$

Finally the results are shown with hsv features and the parameters as shown in Table 1

Table 1: Parameters

Parameter	Description	Value
Radius	Distance below which the similarity is calculated)	15
σI	The sigma of gaussian for the intensity space	0.1
σX	The sigma of the Gaussian for the distance	4

1.2 Output

As I implemented the non recursive one, I got only one segmentation map corresponding to the second lowest eigenvector. For getting more regions, I have clustered the values using $K = 32$ means clustering. For $K = 2$, I was getting meaningless outputs.

1.3 Output Analysis

The segmentation map is quite good for uniformly colored objects as pen whereas in case of aeroplane it segments into two halves as the bottom part has different color.

2 Question 2

The code is available in Ques2 FCN.ipynb. It is based on pytorch hub based models. I have used a pre-trained Fully Connected ResNet50 model for getting the semantic segmentation maps of the images. The model is trained on the COCO 2017 dataset on a subset of Pascal Classes. The output has 21 classes present in the Pascal dataset.
Important Note: We need to use the same transform as done while training to ensure good and valid segmentation outputs.

2.1 Comparison of Ncut and FCN approach

2.1.1 Testing the two approaches

I have used two images from the val dataset of PASCAL VOC and two other images from the internet for testing. This split makes us test on both the seen distribution and the unseen distribution as a test for the performance of the model.

2.1.2 NCut Approach

The Ncut is able to segment the images based on the color information. So it's outputs are very different from a segmentation network (FCN). In this case we see that for a complex image like car image it is unable to segment parts properly whereas for simple shapes like pen, monitor etc., the segmentation maps are better. It performs well on generalized images.

2.1.3 FCN Approach

The FCN approach is quite restrictive to the classes present in the dataset. Doesn't generalize to any generic segmentation map technique. The segmentation maps for the Pascal VOC dataset images i.e. car and aeroplane are really accurate. But in case where there is a pen present and a monitor present it is not able to segment properly and gives spurious predictions for them i.e. predicts presence of a bird and some other class whereas the required output was to classify them as background. (I think we should train the network with just the background images to remove the bias of always predicting something)
Run Time Analysis: The Ncut algorithm needs to be run every time and takes 2-3 minutes to run for each image. Whereas the FCN inference is quite quick.
The images segmented from Ncut and FCN are compared side by side in Fig. [1][2][5][4]

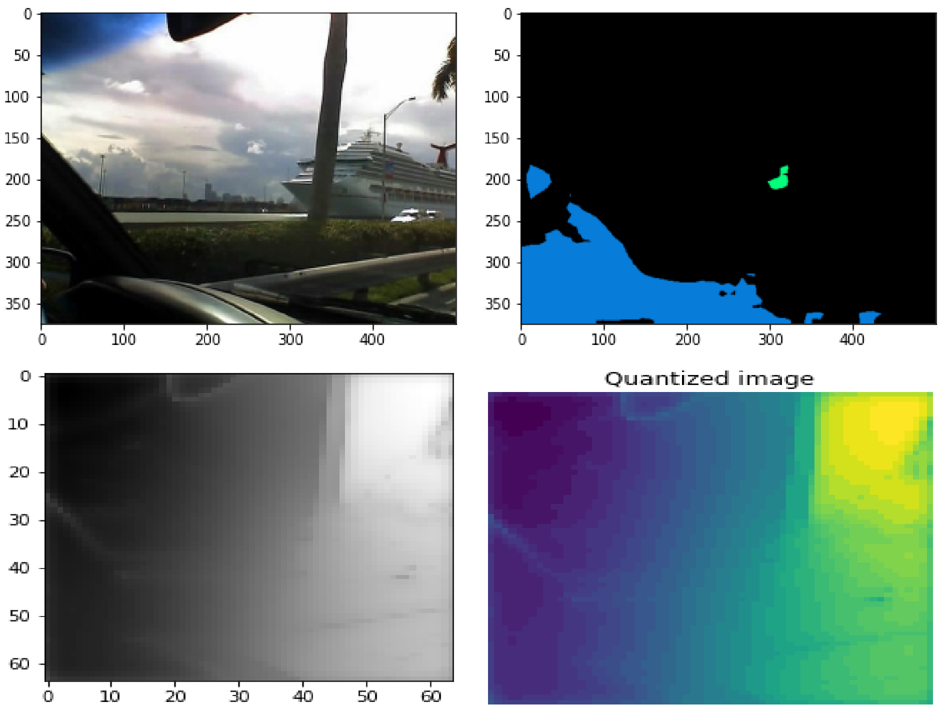

Figure 1: (a) Car (Pascal VOC) (b) FCN Output (c) Grayscale segmentation map from Ncut(Corresponding to second smallest eigenvalue) (d) Quantized image from K means clustering of the segmentation map

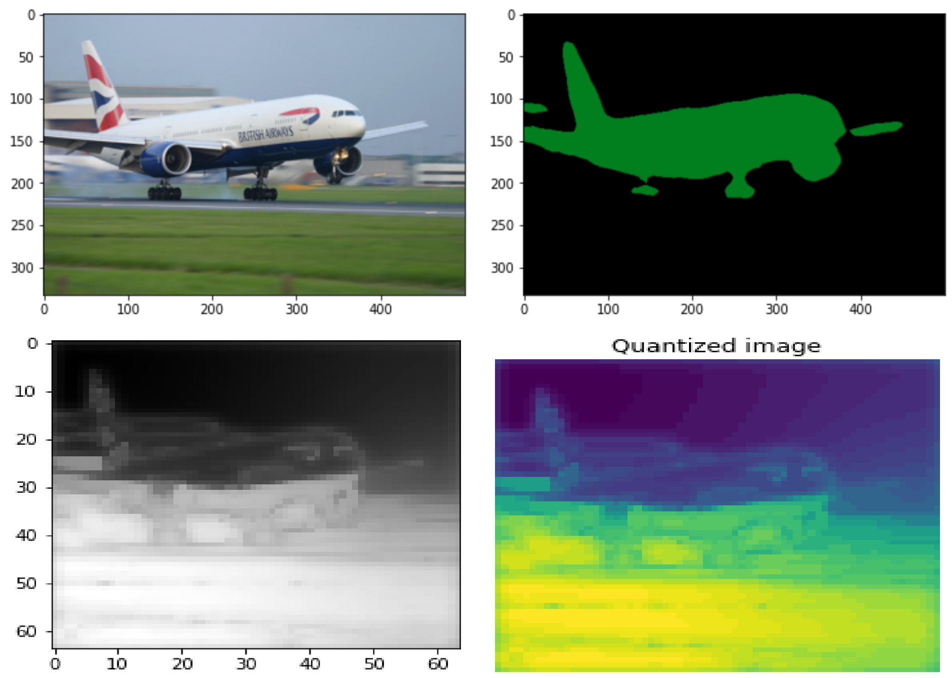

Figure 2: (a) Aeroplane (Pascal VOC) (b) FCN Output (c) Grayscale segmentation map from Ncut(Corresponding to second smallest eigenvalue) (d) Quantized image from K means clustering of the segmentation map

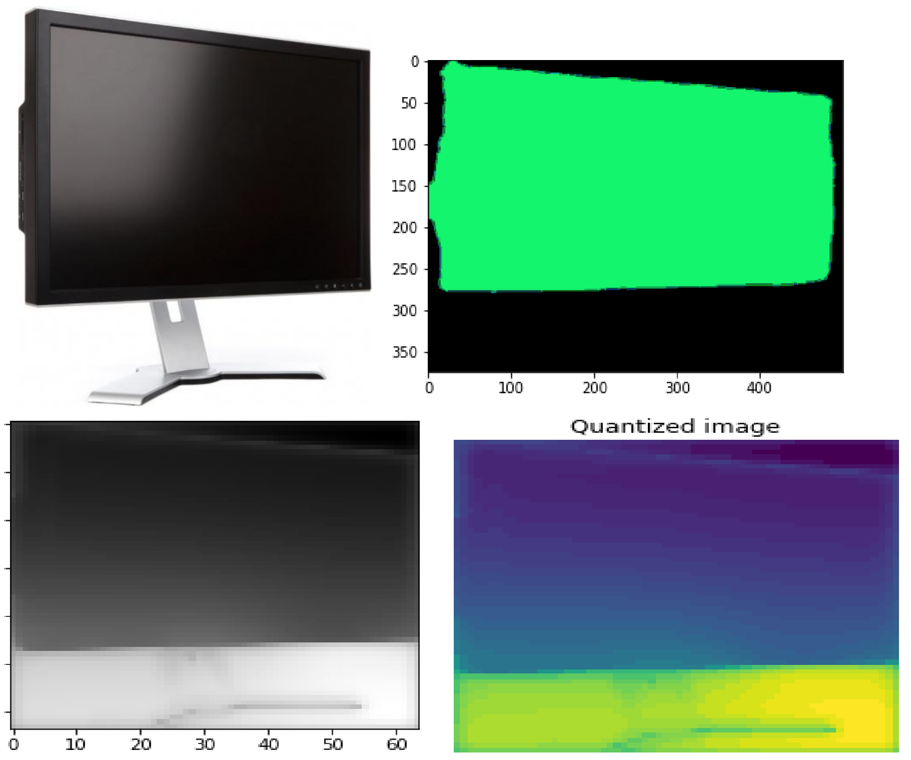

Figure 3: (a) Monitor (Pascal VOC) (b) FCN Output (c) Grayscale segmentation map from Ncut(Corresponding to second smallest eigenvalue) (d) Quantized image from K means clustering of the segmentation map

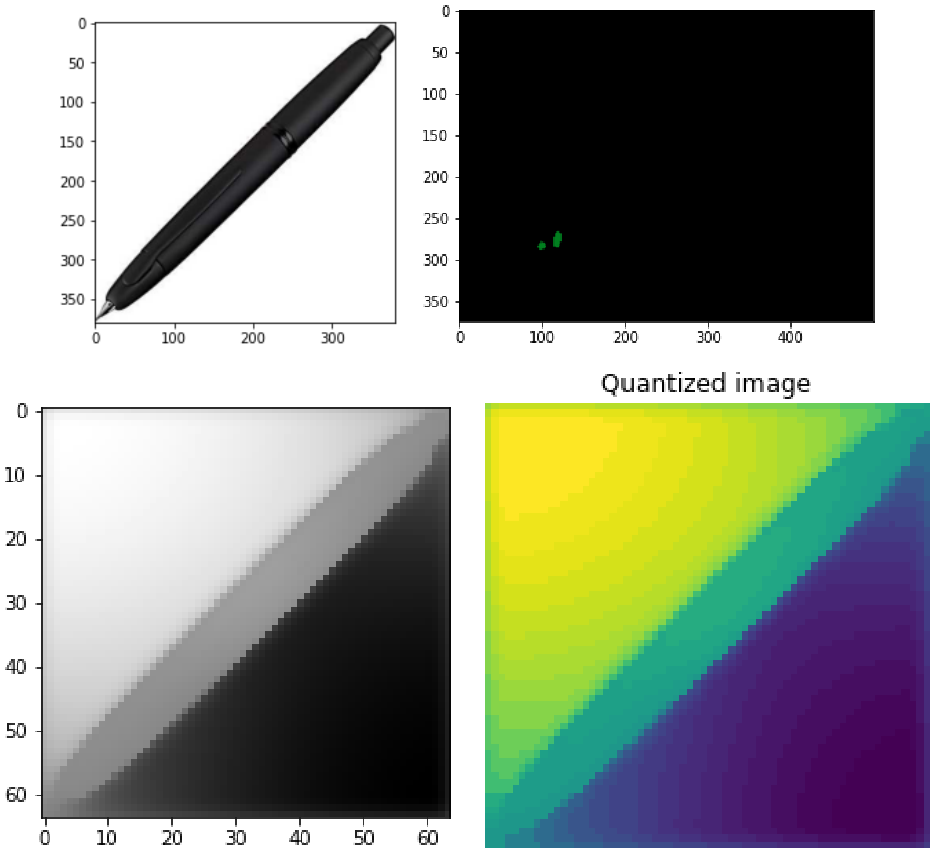

Figure 4: (a) Pen (Pascal VOC) (b) FCN Output (c) Grayscale segmentation map from Ncut(Corresponding to second smallest eigenvalue) (d) Quantized image from K means clustering of the segmentation map

3 Question 3

The source code is available in Ques3.ipynb.

3.1 Data

These models expect a 3-channeled image which is normalized with the Imagenet mean and standard deviation, i.e., $mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]$. All the data is in dataset1 folder. It contains training, testing images and their annotations in different folders. All the images are of [360,480,3] size with 12 classes in each image.

3.2 Model

I have taken a VGG16 pre-trained model and replaced its fully connected layer with a convolution layer of the kernel size 1x1. I have frozen all the weights except for fully connected layers. So as to get the output of the same size as input image, I have added up-sampling layers followed by 2D convtranspose layers. Details of all the hyper -parameters used to train the network is given in Table [2]. The input images are resized to [224, 224]. We have 2D images where each pixel corresponds to a class. We have converted each image into a segmentation map where each pixel is colored according to the class it belongs to. Mean IoU is 0.409 from the trained model. Segmented image for test example is in Fig. [3.2].

Table 2: Hyper Parameters

Loss	Cross Entropy
Optimizer	Stochastic Gradient
Learning Rate	0.001
Momentum	0.9
decay	$5 * *(-4)$
No. of epochs	200

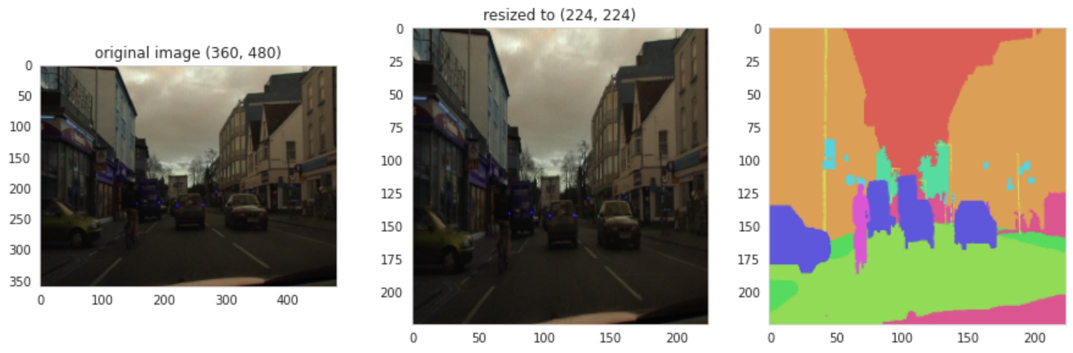

Figure 5: (a) Original (b) Re-sized (b) Segmented Image