Documentation for task-1

By Aravind Sarath Chandran

To get the idea of CNN’s I basically just watched the videos provided in the resources provided

To apply edge detection there are various methods and since we were learning about convolutions I decided to choose that method. In this method I used predefined kernels like sobel,scharr etc by not just importing it from the open cv library but by creating a function to do the entire process.The two methods I used were sobel and scharr , I did give a bit more importance to sobel as it was a bit more common than scharr and personally It worked a bit more better for me.

Firstly I created a function to do the convolution between the image and the kernel,for this I created a function called conv\_transform because for convolution unlike correlation we have to rotate the kernel by 180 degrees.

Secondly I created a function to convolute the image and kernel, for the kernel for both the sobel and scharr operators I separately created matrixes and filled the values(since they were already predefined).i computed the values of the height and width of the image and the kernel and then I found the absolute division of kernel height and width by 2(here both height and width were 3) because of the specific way in which the convolution is done I need to ensure that it was done in the same manner as it needed to be computed.

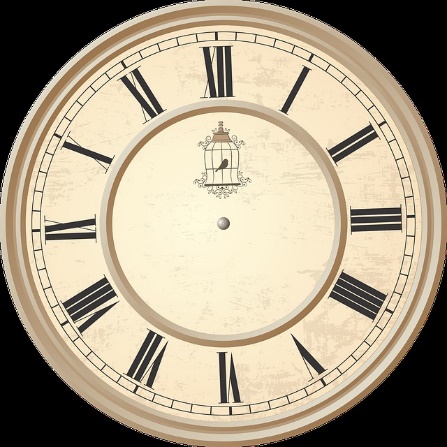
Thirdly I created a matrix representing the values of gaussian filter. Since I had actually added this function at the end I felt that using a 5X5 matrix was better than the 3X3 one so I created the 5X5 gaussian filter kernel and convoluted it to return the image. Here gaussian filter is added since in most cases there will be a lot of unwanted noise that we need to remove before applying edge detection.

Lastly I just needed to either do sobel and scharr , personally for me I felt sobel was a bit more effective in executing than scharr because I did find it a bit difficult on finding different threshold values for different images using the otsu function and I wasn’t able to manipulate it as I wanted to so I stuck to finding a threshold value that works for most images. For the sobel function I easily found it but for the scharr function I could not find a proper value that worked well for all images.

A close up of a flower

Description automatically generated

Original image sobel edge scharr edge

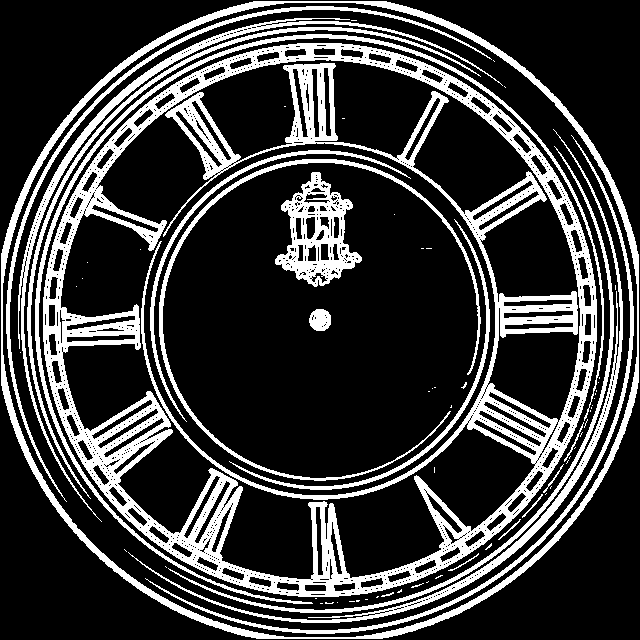
A clock with roman numerals

Description automatically generatedA clock with roman numerals

Description automatically generated

Original image sobel edge scharr edge

Here for the above results I kept the threshold value constant and I felt that the images were better and clearer in sobel edge detection.However i did get very clear images in the scharr method but I felt that I needed to change the threshold .For example in the image of clock I did get the below image through scharr method, the image is very clear,but I did have to change the threshold value and this threshold value did not give a very good image of the flower.



This is all I had done to complete task 1. I had used a vast amount of resources ranging from youtube videos on how to write the code to reading materials about each library and also read a bit about gaussian,sobel,scharr,otsu etc.