REPORT

This is a brief summary of the work up to the current stage of Text Detection, that was assigned to us for the internship under BOOKSERB.

1. Collected 16 images (front and back cover of books) in order to make the dataset and added them into a folder.

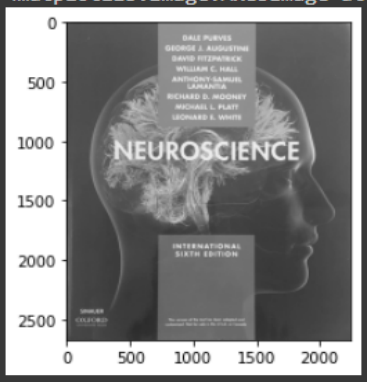
2. Created Dataset :
   2.1 Imported the necessary packages including open cv (by importing cv2) and glob. Glob is used to list or return file paths for the specified patterns.
   2.2 Read multiple images from the folder, glob module returns the list of files with their full path.
   2.3 A dictionary of the image paths and their corresponding indexes are created.
   2.4 Dataframe is created using the dictionary .
   2.5 The required dataset is obtained by converting the Dataframe into a csv file.

3. Text detection using pytesseract:
   By importing the necessary packages and by reading the image path, each text in the image is detected and returned by using the pytesseract module. But due to some defects like lightning, blurred images etc… some texts are not detected and also some are detected incorrectly.
   So, there is a need to pre-process the image.

4. Image preprocessing :
   In this step, the main objective is to make it as easy as possible for the system to distinguish a character or word from the background.
   We did the following steps:
   4.1 converted the images into grayscale image :
   So that the images displays only the darkest balck to the brightest white colors ie., the image contains only balck, white and gray colors.

```
[ ]  img_gray=cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
     plt.imshow(img_gray,cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f18bd931950>
```

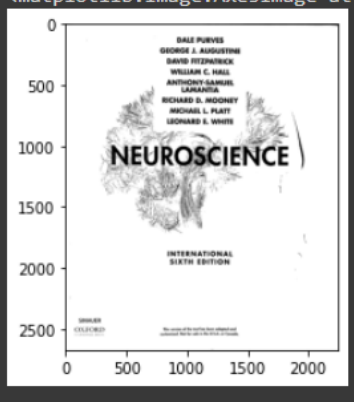


4.3 Thresholding:

A type of image segmentation , where we changed the pixels of the image to make the image easier to analyze ie., by converting the grayscale image into a binary image ie., simply black and white.

```
[ ]  rat,thresh=cv2.threshold(img_gray,200,255,cv2.THRESH_BINARY_INV,cv2.THRESH_OTSU)
     plt.imshow(thresh,cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f18bd50d8d0>
```

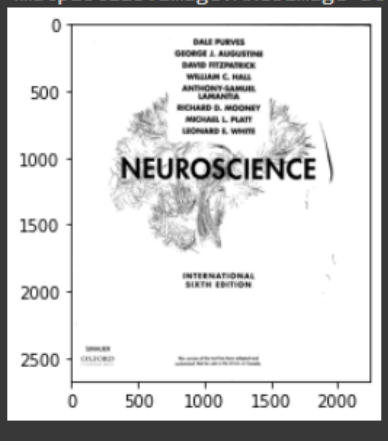


4.2 Gaussian Blur:

Hence removed the random noise from the image and also it is used to make the Image smooth in which edges are not observed.

```
[ ]  blur = cv2.GaussianBlur(thresh, (3,3), 0)
     plt.imshow(blur,cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f18bd473750>
```



The link for the notebook of the code showing the above procedures is given below:
https://colab.research.google.com/drive/191WKq4v6uSvn0_fh4Q-7DR_I7_SbOmC_?usp=sharing

BY,
FAZNA FATHIMA SHAJU
SWALIHA M,
M.SC DA