E9 241 Digital Image Processing

Assignment 1 Report

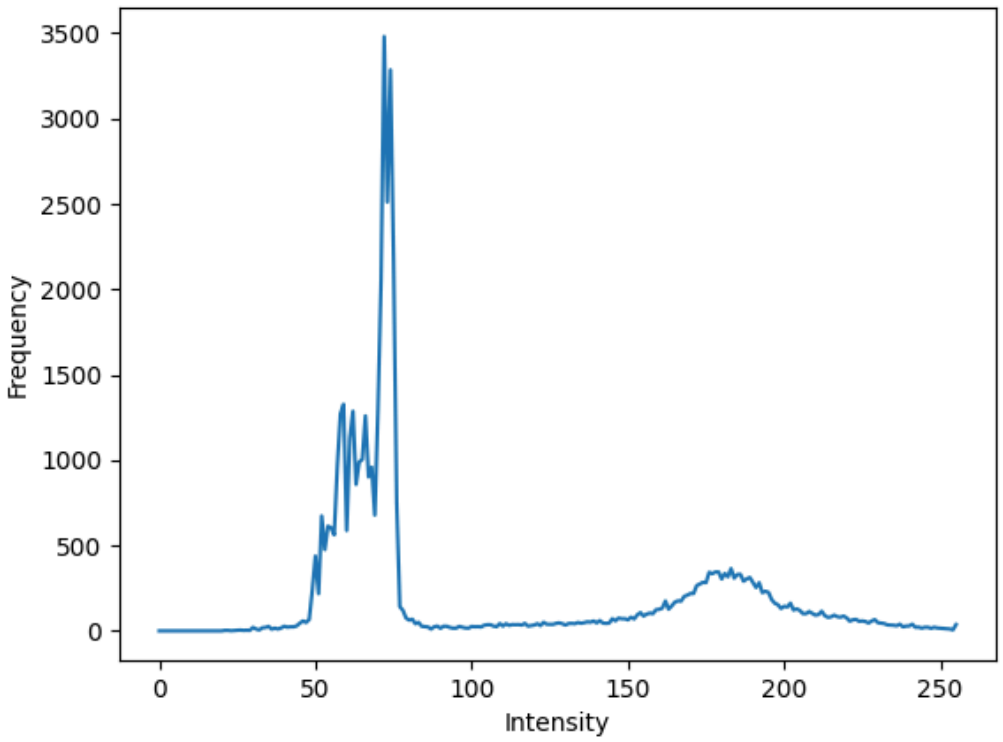
Prepared by Indranil Patra, indranilp@iisc.ac.in

Things to note for running the code:

a. All images are stored in the "./images" folder under root folder. Hence the file path should start with "./images/…"
b. The entire code along with the driver code is in the code.py file
c. All the plots in the report are present as separate .png files in the folder.

1. Histogram computation involves two steps. First, computation of frequencies of each intensity level. This is computed and stored in `freq` variable. Then dividing each value in `freq` array with the size of the image $m*n$.

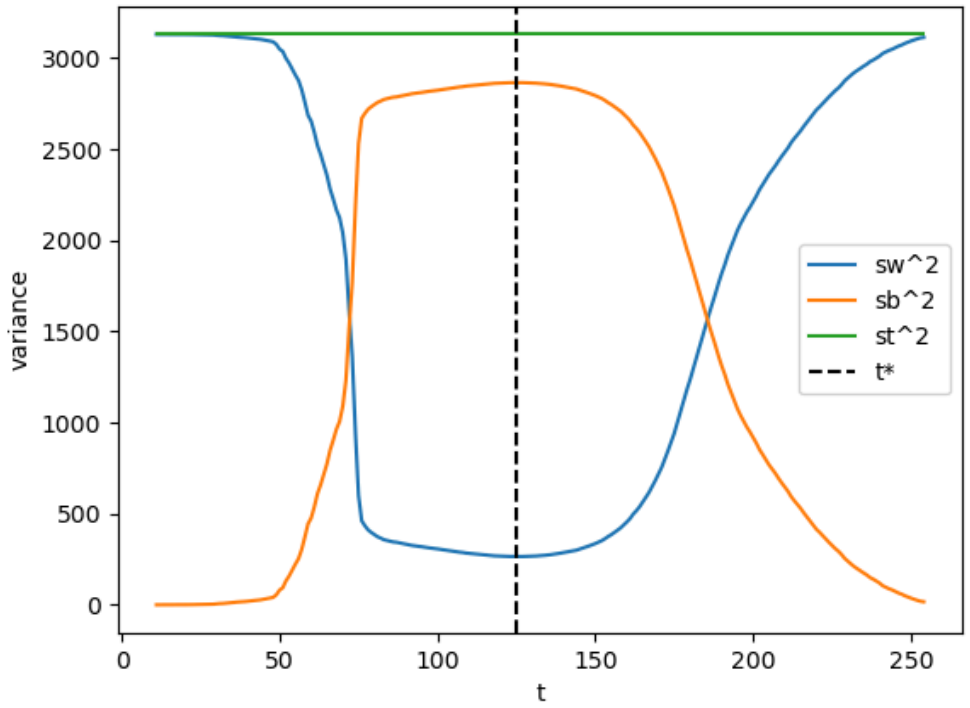
   The average intensity computed as $\sum_{k=0}^{256} k \times p(k)$. This is verified with the actual average of the image computed as `np.mean(img)`. Both the methods gave the same results.



Histogram of the intensities

From the histogram, it is evident that lower grey intensity values are significantly more than higher intensity values. There are two peaks, a very sharp peak around 80 and another shorter peak around 175. Hence, a threshold between these two values would be the optimal threshold for binarization.

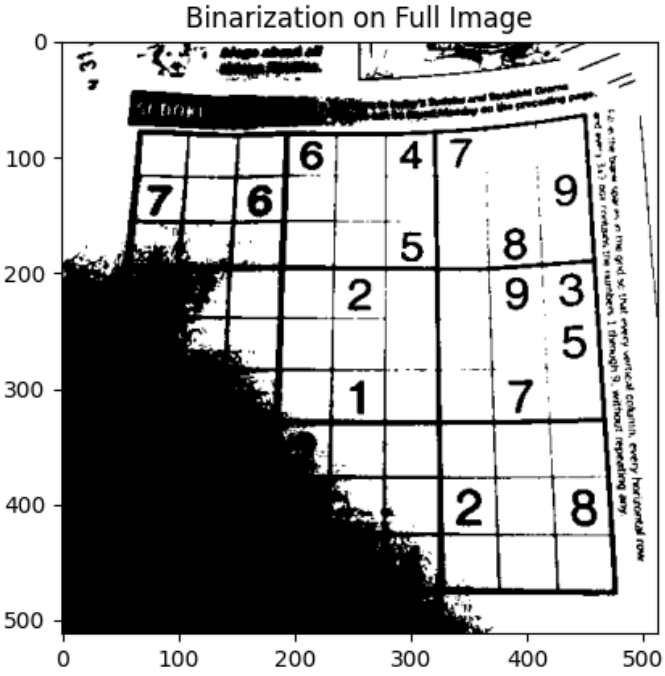
2. The t got after minimising within class variance is **125** which is the same value got after maximising between class variance. Hence it is verified that both the methods are equivalent.

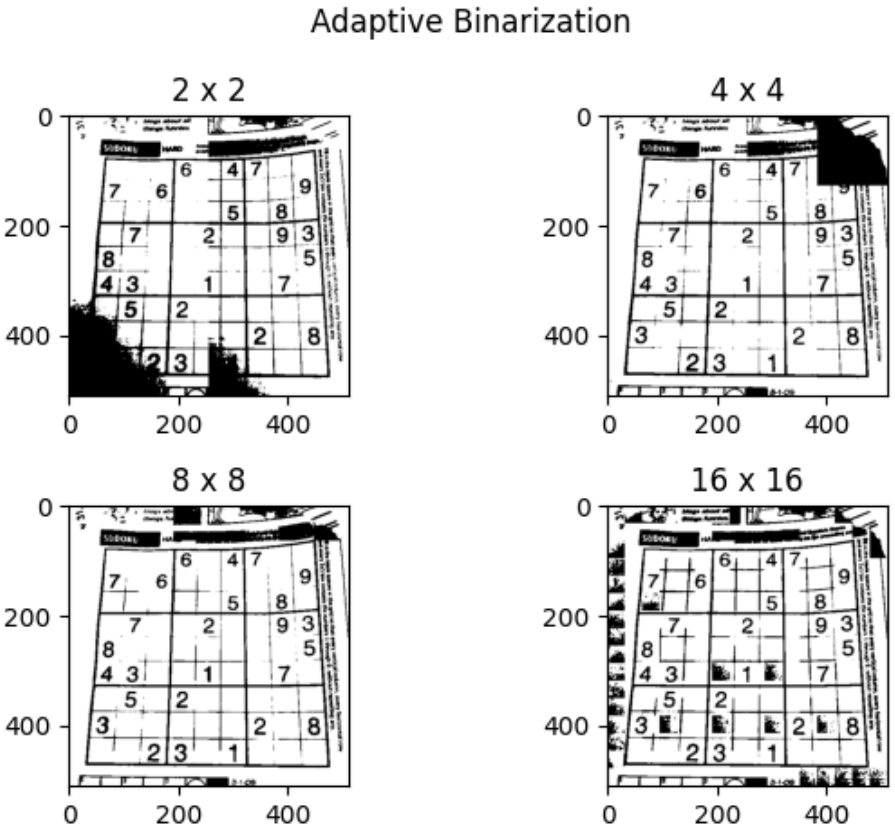


Variance vs t plot

In the legend sw^2 corresponds to σ_w^2, sb^2 corresponds to σ_b^2 and st^2 corresponds to σ_T^2. From the graph, it is verified that $\sigma_w^2 + \sigma_b^2 = \sigma_T^2$.
t * corresponds to the optimal threshold.

3.



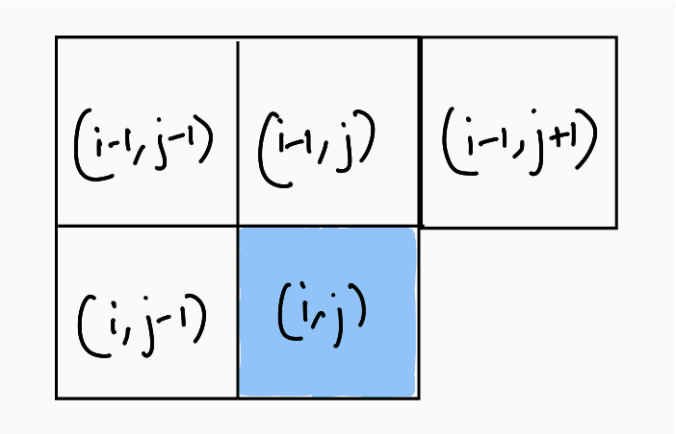


Binarization taking the entire image as one results in the bottom left portion of the image getting completely noisy.

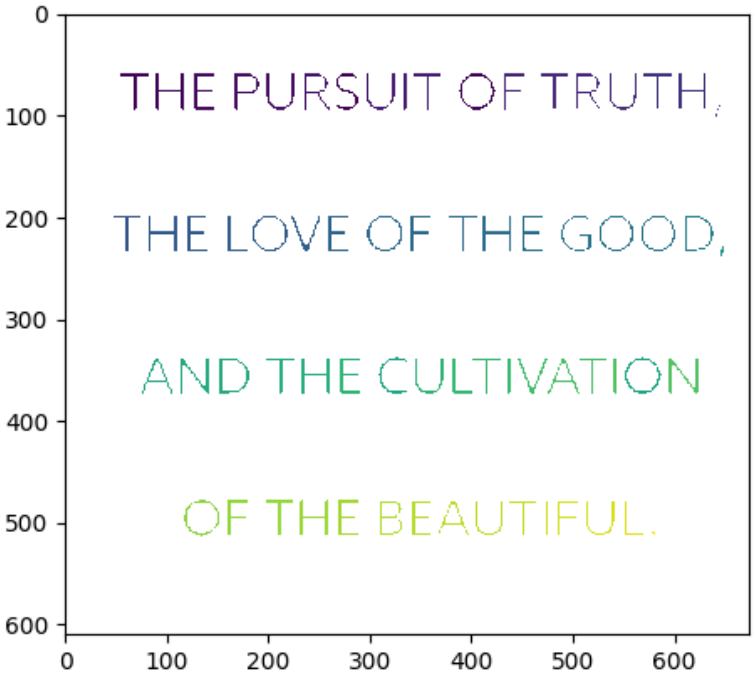
On the other hand, adaptive binarization taking 2×2 blocks yield a better result. With an 8×8 blocks, there is better local thresholding, there is accurate

segmentation of background and foreground. In case of 16×16 blocks due to low block size, there is over-segmentation and boundary artefacts become an issue.

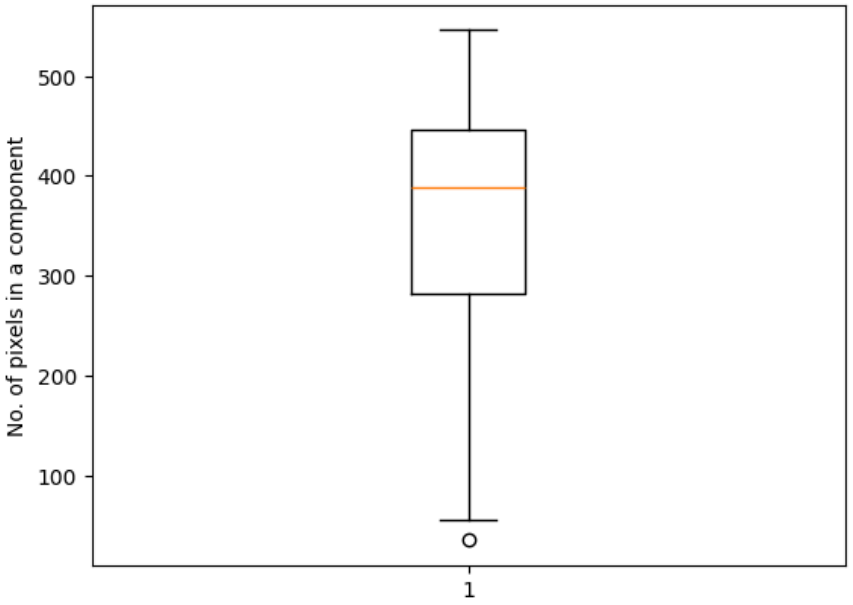
4. The number of connected components = 67 and the number of characters excluding the punctuations = 64.



Every pixel is checked with these neighbours. If any one them has pixel value 0, then $I(i, j)$ is part of the same component.
For the cases $(i - 1, j)$ and $(i - 1, j + 1)$, whenever the components of those pixels change, we need to cascade those changes to all the pixels which were part of the same component.
For the case none of the pixels in the checking window has value 1, a new component is assigned to pixel (i, j).



Heatmap of the connected components



Box plot of the no. of pixels belonging a component

Observing the box plot of the number of pixels in a component, and calculating the quartile-2 and quartile-1, IQR, if the number of pixels in a component is less than $Q1 - IQR$, then it is probably punctuation. Using this, we get 3 punctuations in the image.