Assignment Project Report
Adaptive Thresholding: Edge Detection in Images

Name: Pavan Tiwari

Course: AI and ML
(Batch 4)

- **Problem Statement**

Using OpenCV, first convert any image with varying High condition to a grayscale image. Now implement edge detection first using the canny edge detection. Then apply simple thresholding and also Adaptive/OTSU thresholding using OpenCV to see the working of each of these methods. Once you obtain good results, use the obtained edge detection result as a mask to give color to all the edges (if edges use the color from the original image, else leave it black only).Prerequisites

- Software:
  - Python 3 (Use anaconda as your python distributor as well)

- Tools:
  - Pandas
  - Numpy
  - Matplotlib
  - Seaborn
  - OpenCv

- **Method Used**

Edges define the boundaries between different regions in an image, which helps in matching the pattern, segment, and recognize an object. In simple thresholding, the threshold value is global, which is prone to fail in many cases. Adaptive thresholding is a modified method where the threshold value is calculated for each pixel based on a smaller region around it. Therefore, there will be different threshold values for different regions which gives better results for images with varying illumination

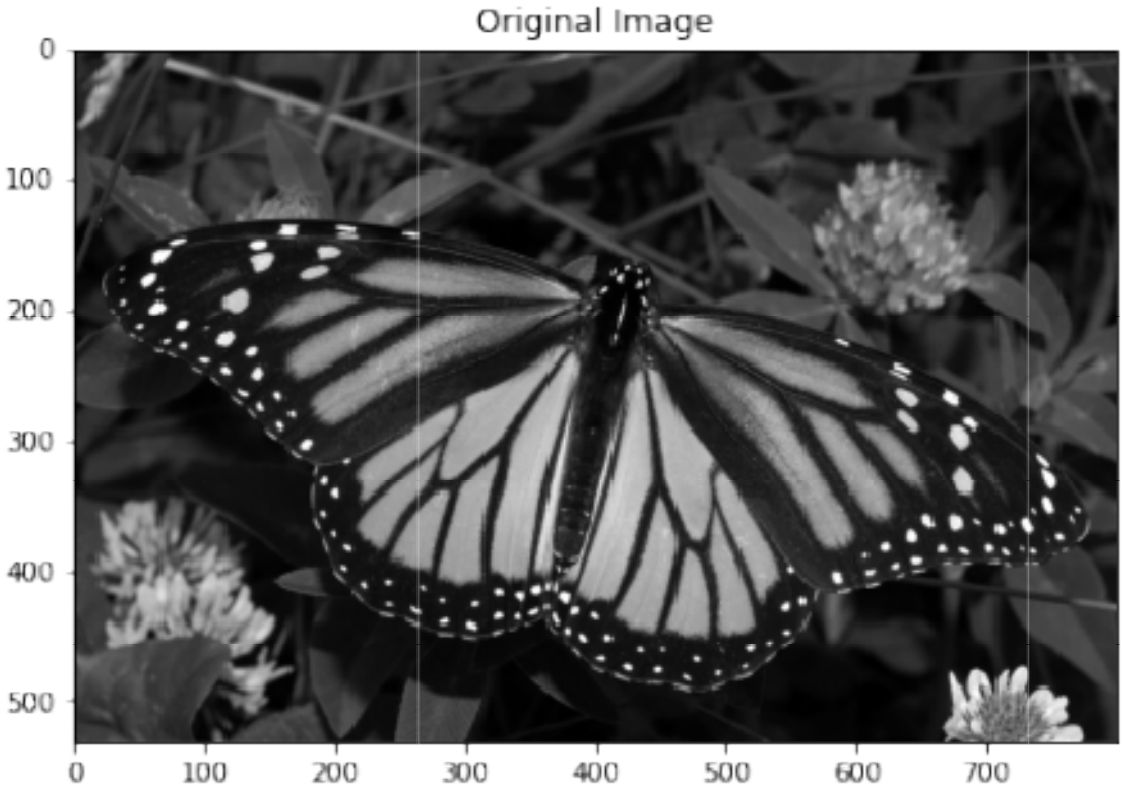
- **Implementation:**

1. Load all required libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
sns.set_style('whitegrid')
plt.style.use("fivethirtyeight")
%matplotlib inline
```

```
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
train.head()
```

2. Visualizing data



3. Applying Thresholding On Image

Edge Detection using Adaptive Thresh holding Last Checkpoint: 26/07/2021 (autosaved)

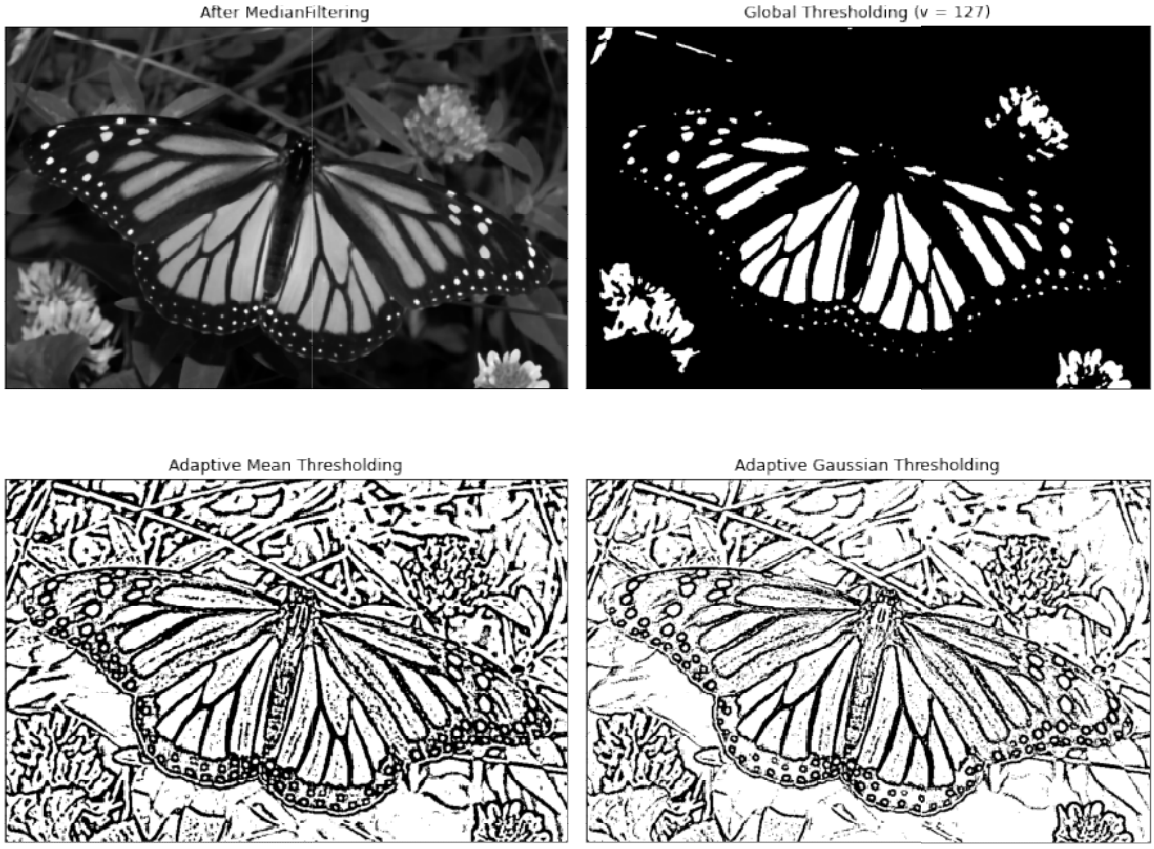
View Insert Cell Kernel Widgets Help Trusted Pyth

Run Markdown

```
img = cv2.medianBlur(org_img_01,5) ## Remove noise using median Blur

## 2nd argument - threshold, 3rd Argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

## threshold value is the mean of neighbourhood area
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BINARY,9,2)

## threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,9,2)

titles = ['After MedianFiltering','Global Thresholding (v = 127)',
            'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']

images = [img, th1, th2, th3]

plt.figure(figsize=(12,10))
for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.tight_layout()
plt.show()
```

Activate Windows
Go to Settings to activate W

4. Output After Thresholding



Otsu Method

Edge Detection using Adaptive Thresh holding Last Checkpoint: 26/07/2021 (autosaved)

View Insert Cell Kernel Widgets Help Trusted Pyth

Run Markdown

```
img = cv2.medianBlur(org_img_01,5) ## Remove noise using median Blur

## 2nd argument - threshold, 3rd Argument - Value assigned if pixel is greater than threshold
ret,th1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

## threshold value is the mean of neighbourhood area
th2 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_MEAN_C,cv2.THRESH_BINARY,9,2)

## threshold value is the weighted sum of neighbourhood values where weights are a gaussian window.
th3 = cv2.adaptiveThreshold(img,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,9,2)

titles = ['After MedianFiltering','Global Thresholding (v = 127)',
          'Adaptive Mean Thresholding', 'Adaptive Gaussian Thresholding']

images = [img, th1, th2, th3]

plt.figure(figsize=(12,10))
for i in range(4):
    plt.subplot(2,2,i+1)
    plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.tight_layout()
plt.show()
```

Activate Windows
Go to Settings to activate V

5. Result After Otsu Method

Image After Median Filtering

Image after Otsu's Thresholding