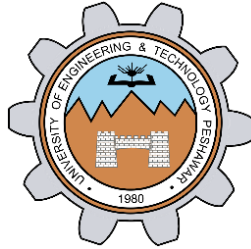


# Assignment 01



Spring 2024

## Digital Image Processing

Submitted by:

**Safi Ullah Khan (20pwcse1943)**

Class Section: **B**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

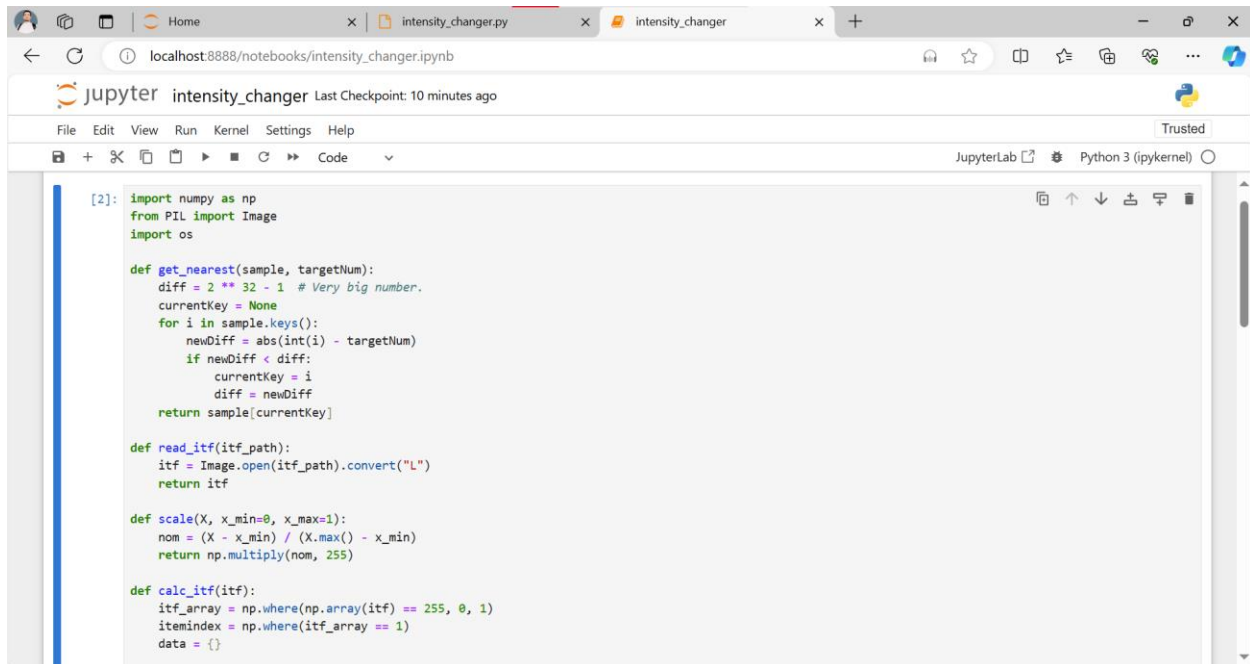
Submitted to:

Dr. Abeer Irfan

(May 24, 2024)

**Department: Computer System Engineering**

## Python code



The screenshot shows a JupyterLab notebook titled 'intensity\_changer' running on a local host. The code defines several functions for image processing:

```
[2]: import numpy as np
from PIL import Image
import os

def get_nearest(sample, targetNum):
    diff = 2 ** 32 - 1 # Very big number.
    currentKey = None
    for i in sample.keys():
        newDiff = abs(int(i) - targetNum)
        if newDiff < diff:
            currentKey = i
            diff = newDiff
    return sample[currentKey]

def read_itf(itf_path):
    itf = Image.open(itf_path).convert("L")
    return itf

def scale(X, x_min=0, x_max=1):
    nom = (X - x_min) / (X.max() - x_min)
    return np.multiply(nom, 255)

def calc_itf(itf):
    itf_array = np.where(np.array(itf) == 255, 0, 1)
    itemindex = np.where(itf_array == 1)
    data = {}
```

```
def calc_itf(itf):
    itf_array = np.where(np.array(itf) == 255, 0, 1)
    itemindex = np.where(itf_array == 1)
    data = {}

    for i in range(len(itemindex[0])):
        data[itemindex[1][i]] = float(itemindex[0][i])

    return data






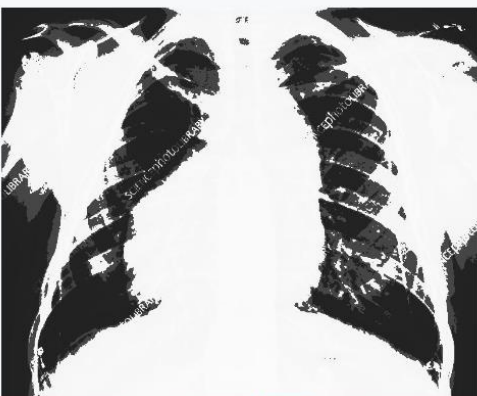
def intensity_transformation(image_path, itf_path):
    image = Image.open(image_path).convert('L')
    itf = read_itf(itf_path)
    data = calc_itf(itf)

    color = np.array(image)
    new_color = np.zeros_like(color)

    for col in range(color.shape[0]):
        for row in range(color.shape[1]):
            try:
                new_color[col, row] = data[color[col, row]]
            except KeyError:
                new_color[col, row] = get_nearest(data, color[col, row])

    new_image = Image.fromarray(new_color).convert('L')
    new_image.save(f'output/modified_{os.path.basename(itf_path).split(".")[0]}_{os.path.basename(image_path)}')
    new_image.show()
```

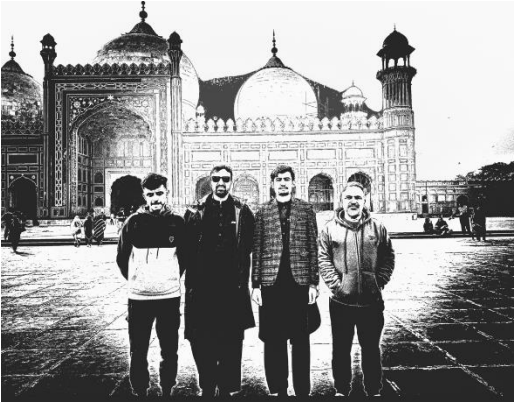
Before after intensities transformations:

ORIGINAL	ITF	RESULT
		
		

Before



after



Before



after

