

21F-9237

Zabar Zeeshan

Computer Vision - Assignment 1

Assignment Report: Real-world Image Enhancement and Analysis

1. Introduction (The Problem)

X-ray images are often low-contrast, which makes it difficult for radiologists and medical AI systems to detect subtle details such as fractures, small lesions, or tissue abnormalities. Enhancing these images improves visibility and can support more accurate diagnosis.

2. Dataset

Source: Publicly available X-ray dataset (e.g., chest X-ray images).

Format: PNG/JPG images, grayscale.

Characteristics: Many images appear dark, with important structures hidden due to poor contrast.

Problem to Solve: Improve the contrast and visibility of medical structures in X-ray images.

3. Methodology & Justification

Technique 1: Histogram Equalization

Why Chosen?

X-rays often have pixel intensities clustered in a small range (low contrast). Histogram Equalization redistributes pixel intensities across the full range [0–255], making darker and lighter regions more distinguishable.

Transformation Function: Built-in OpenCV function `cv2.equalizeHist(img)`

Technique 2: Gamma Correction (Power-Law Transformation)

Why Chosen?

Medical images often need selective brightness enhancement. Gamma correction allows us to brighten shadow regions without over-exposing bright regions.

Transformation Function: $s=c \cdot r^\gamma$

4. Results & Analysis

Visual Comparison

- Histogram Equalization: Improved global contrast. Bone structures became more defined.
- Gamma Correction: Highlighted hidden details in darker areas (lungs, tissues).

Histogram Analysis

- Original: Pixel values clustered in mid-range (low contrast).
- After Hist Eq: Histogram spread out across full range → improved contrast.
- After Gamma Correction: Histogram shifted → darker regions became brighter, more visible.

Critical Evaluation

- Histogram Equalization worked better for global contrast.
- Gamma Correction worked better for revealing hidden structures in shadows.
- A combination could be tried, but for medical images, too much enhancement may introduce artifacts.

5. Conclusion

Most Effective Technique: Histogram Equalization for global improvement, with Gamma Correction for local detail enhancement.

Final Recommendation: Use Histogram Equalization for overall preprocessing of X-ray datasets, and Gamma Correction selectively for images with hidden details in dark areas.

6. Appendix (Code)

```
# Histogram Equalization
```

```
hist_eq = cv2.equalizeHist(img)
```

```
# Gamma Correction
```

```
gamma_img = gamma_correction(img, gamma=0.5)
```

```
# Display results
```

```
show_images(img, hist_eq, "Original", "Histogram Equalized")
```

```
show_images(img, gamma_img, "Original", "Gamma Corrected")
```

Complete Code(Cell by Cell):

1. Mounting of Dataset:

```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
import os

dataset_path = "/content/drive/MyDrive/xray_dataset/" |
print("Files in dataset:", os.listdir(dataset_path)[:5]) # Show first 5 files
```

2. Files in dataset: ['IM-0015-0001.jpeg', 'IM-0009-0001.jpeg', 'IM-0005-0001.jpeg', 'IM-0010-0001.jpeg']

3. Loading Image:

```
import cv2
import matplotlib.pyplot as plt

# Load first image in dataset
img_path = os.path.join(dataset_path, os.listdir(dataset_path)[4])
img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)

plt.imshow(img, cmap='gray')
plt.title("Original X-ray")
plt.axis("off")
plt.show()
```

Original X-ray



4. Apply Histogram Equalization:

```
import numpy as np
```

```
# Show images side by side
```

```
def show_images(original, enhanced, title1="Original", title2="Enhanced"):
```

```
    plt.figure(figsize=(10,5))
```

```
    plt.subplot(1,2,1)
```

```
    plt.imshow(original, cmap='gray')
```

```
    plt.title(title1)
```

```
    plt.axis('off')
```

```
    plt.subplot(1,2,2)
```

```
    plt.imshow(enhanced, cmap='gray')
```

```
    plt.title(title2)
```

```
    plt.axis('off')
```

```
    plt.show()
```

```
# Show histograms
```

```
def show_histograms(original, enhanced):
```

```
    plt.figure(figsize=(10,5))
```

```
    plt.subplot(1,2,1)
```

```
    plt.hist(original.ravel(), bins=256, range=[0,256])
```

```
    plt.title("Original Histogram")
```

```
    plt.subplot(1,2,2)
```

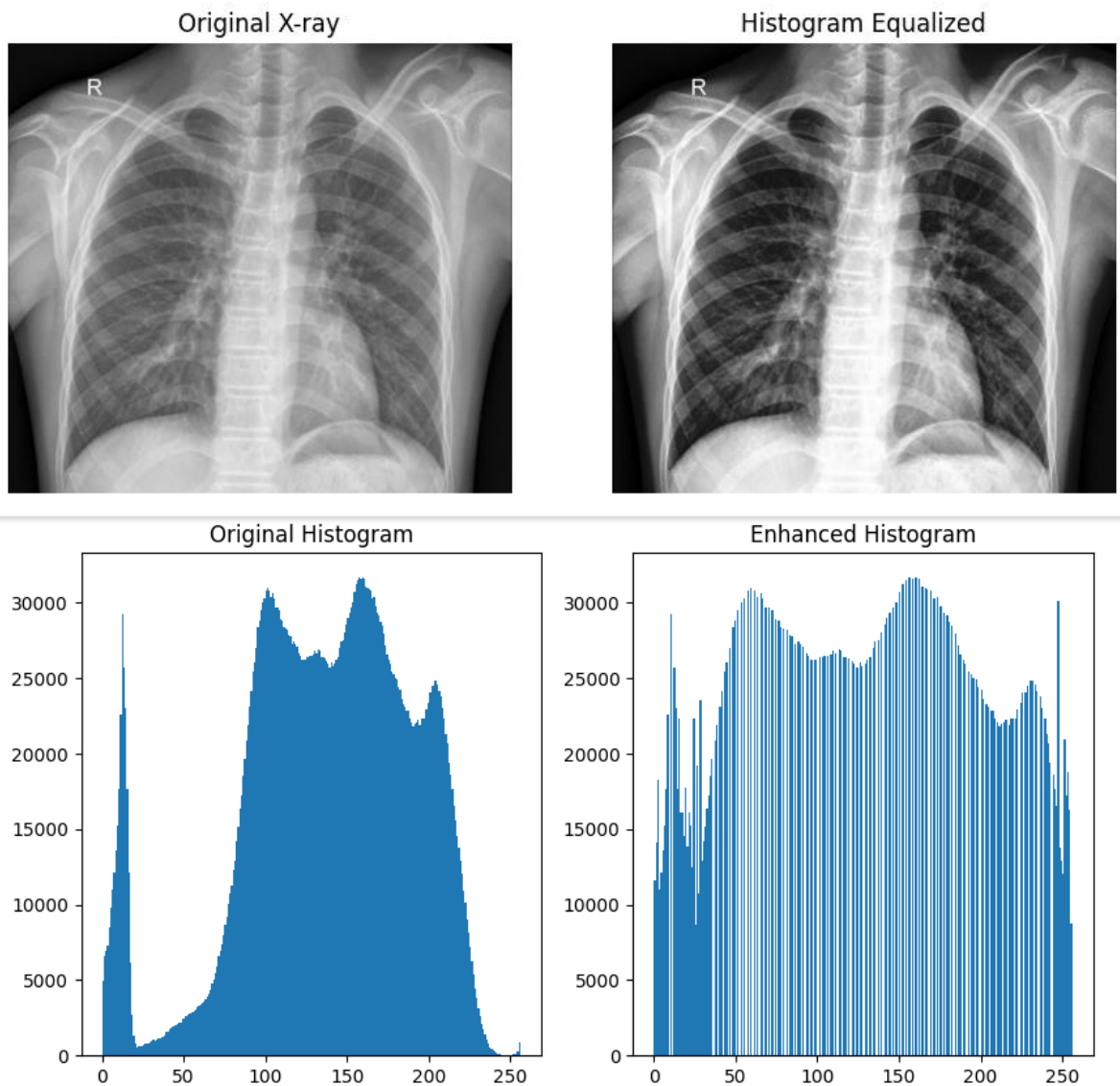
```
    plt.hist(enhanced.ravel(), bins=256, range=[0,256])
```

```

plt.title("Enhanced Histogram")
plt.show()
# Histogram Equalization
hist_eq = cv2.equalizeHist(img)

show_images(img, hist_eq, "Original X-ray", "Histogram Equalized")
show_histograms(img, hist_eq)

```



5. Apply Gemma Correction:

```

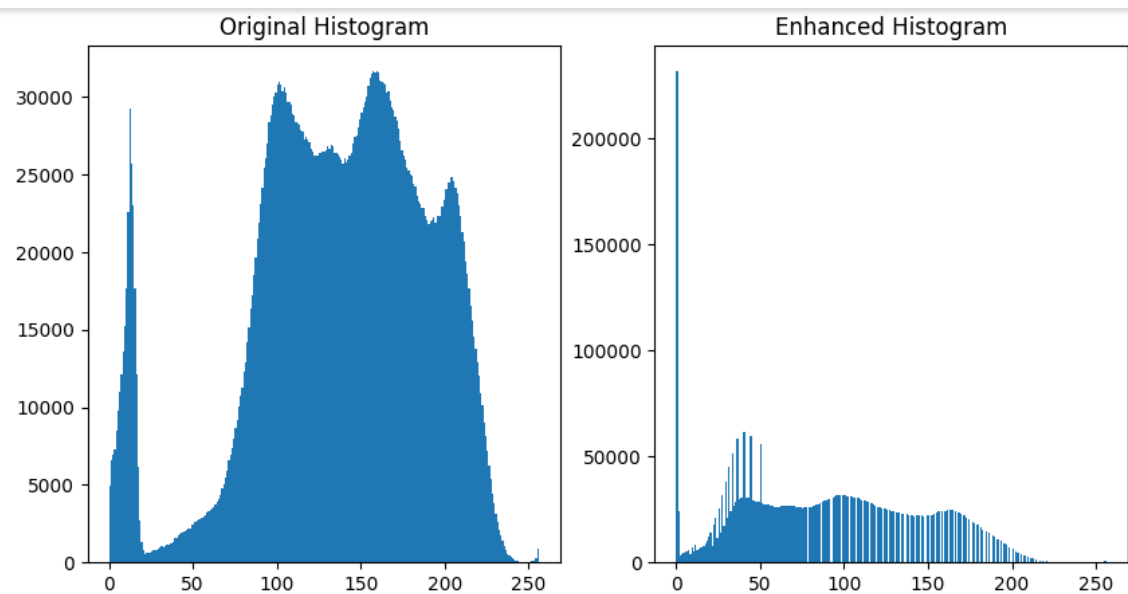
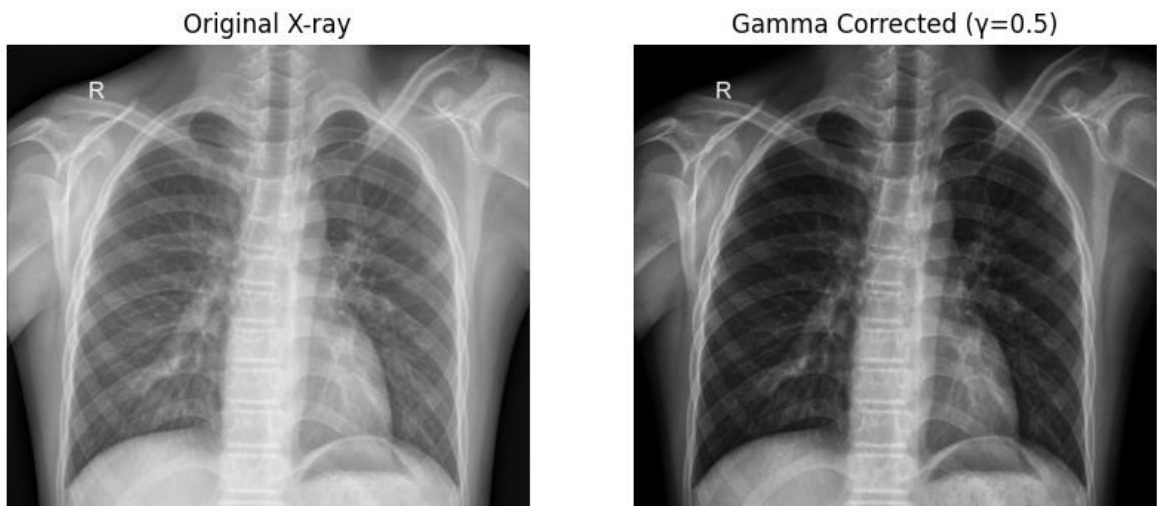
# Gamma Correction
def gamma_correction(image, gamma=0.5):
    invGamma = 1.0 / gamma
    table = np.array([((i / 255.0) ** invGamma) * 255
                      for i in np.arange(0, 256)]).astype("uint8")
    return cv2.LUT(image, table)

```

```
gamma_img = gamma_correction(img, gamma=0.5)
```

```
show_images(img, gamma_img, "Original X-ray", "Gamma Corrected ( $\gamma=0.5$ )")
```

```
show_histograms(img, gamma_img)
```



6. Final Output:

```
# Apply Gamma Correction (point processing)
```

```
gamma_img = gamma_correction(img, gamma=0.5)
```

```
# Show only the gamma corrected image
```

```
plt.imshow(gamma_img, cmap='gray')
```

```
plt.title("Output Image after Histogram Equalization and Gamma Correction")
```

```
plt.axis("off")
```

```
plt.show()
```

Output Image after Histogram Equalization and Gamma Correction

