ERROR ANALYSIS USING JPEG ALGORITHM

Image Processing J Component
Vishwanathan P

Advaith Chandra Srivastav (20BCE0983)
Sanjitha Rajesh (20BCE2541)
Rujula Nitin Patil (20BCE2789)
Lokesh Reddy (20BCB0084)

INITIAL FUNCTION USED

Peak Signal to Noise Ratio (PSNR)

PSNR is a commonly used metric to evaluate the quality of a reconstructed signal or image, compared to its original or reference signal. It measures the difference between the original signal and the reconstructed signal. The higher the PSNR value, the better the quality of the reconstructed signal is considered to be.

Formulae:

PSNR = 10 * log10((MAX^2) / MSE)

PSNR = 20 * log10 (Peak signal value / RMS error)

However, we found a few limitations with PSNR, such as the fact that it is not sensitive to perceptual differences like contrast, texture, sharpness etc. In addition to this, PSNR only considers the difference in pixel values between the original and distorted image, which means it is not efficient in quality evaluation with complex textures or colors.

INITIAL FUNCTION USED (Code)

```
def get_mse(prediction, target):
    return np.sqrt(((prediction-target)**2).mean())

def get_psnr(mse):
    return 20*np.log10(255/np.sqrt(mse))
```

UPDATED FUNCTION

Structural Similarity Index Measure (SSIM)

SSIM (Structural Similarity Index Measure) is a metric that compares the structural similarity between two images. The formula for SSIM is:

$SSIM(x, y) = [l(x, y)^\alpha] * [c(x, y)^\beta] * [s(x, y)^\gamma]$

where x and y are the two images being compared, and l(x, y), c(x, y), and s(x, y) are individual components that measure the similarity in luminance, contrast, and structure, respectively. The constants α, β, and γ weight the relative importance of each component.

The luminance component l(x, y) measures the similarity in brightness between the two images, while the contrast component c(x, y) measures the similarity in contrast. The structure component s(x, y) measures the similarity in structural information between the two images.

The components are computed using the mean, standard deviation, and cross-covariance between the two images, along with small constants added to avoid division by zero. The SSIM score ranges between -1 and 1, with 1 representing perfect similarity.

In general, adjusting the values of α, β, and γ can emphasize the importance of certain components in the SSIM score, depending on the application.

UPDATED FUNCTION (Code)

```
def get_ssim(prediction, target):
    gray1 = prediction
    gray2 = target
    # Constants for SSIM calculation
    K1 = 0.01
    K2 = 0.03
    L = 255

    # Compute the mean, variance, and covariance of the two images
    mu1 = cv2.mean(gray1)[0]
    mu2 = cv2.mean(gray2)[0]
    sigma1 = cv2.meanStdDev(gray1)[1][0][0]
    sigma2 = cv2.meanStdDev(gray2)[1][0][0]
    sigma12 = np.cov(gray1.flatten(), gray2.flatten())[0][1]

    # Constants for SSIM calculation
    C1 = (K1 * L) ** 2
    C2 = (K2 * L) ** 2

    # Compute SSIM
    numerator = (2 * mu1 * mu2 + C1) * (2 * sigma12 + C2)
    denominator = (mu1 ** 2 + mu2 ** 2 + C1) * (sigma1 ** 2 + sigma2 ** 2 + C2)
    ssim_score = numerator / denominator
    return ssim_score
```