SmallSEOTools

PLAGIARISM SCAN REPORT

Words	343	Date	June 21,2021
Characters	2247	Excluded URL	

0%	100%	0	16
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

Content Checked For Plagiarism

8.1 DATASET AND EXPERIMENTAL SETUP
Initially, we started by scraping the data off Hubble Legacy Archive. Using puppeteer(headless chrome), we scraped off hundreds of thousands of colorized images it has available. The Hubble Legacy archive is slow and produces grainy images with lots of noise and unprocessed images. A filter for M101 (Messier 101) galaxy rendered more than 80 thousand images with a 1 degree difference between consecutive right ascension. The data is large and has no particularly efficient way to clean without human investment. Cleaning tens of thousands of images by handpicking noiseless and well colored images is time consuming. For training the SRGAN, we need high resolution, well colored images.
Consequently, we scraped the Hubble Heritage project instead. The Hubble Heritage project releases the highest-quality astronomical images. They are all stitched together, colorized and processed to eliminate noise. Hubble Heritage then selects the best, most striking of these for public release. However, there are only 150 of these images that are actually useful. We scraped images from the main Hubble website as well so as to increase the amount of data we had. This provided an extra approximately 1000 images. Each image is a JPG image with dimensions of 256256 pixels and contains 3 channels of RGB.
We use a mini-batch gradient descent with a batch size of 10, 16 and 32 for different iterations. We use Adam optimizer with b1 = 0:5 and b2 = 0:999. The generator and discriminator has a learning rate of 2e□4 which remains constant throughout the training. We train the model with different epochs ranging between 20,50 and 100, saving the best model weights determined by L1 norm between the output of the generator and the target image. We use early stopping with a patience value of 10. The image size is 6464. Our implementation is using Python, numpy, Tensorflow and tf-keras. It takes about 24 hours to complete training over the COCO dataset and about 12 to 13 hours to fit the model on given astronomical data on a NVIDIA Tesla K80 GPU.

Sources	Similarity