Image Colourization using WGANs

CS726 Project - Team Blurons

Deepak Thorat (190020037)
Mihir Nandawat (190020071)
Satwik Murarka (190020101)
Rhythm Shah (190110074)

Table of Contents

Introduction

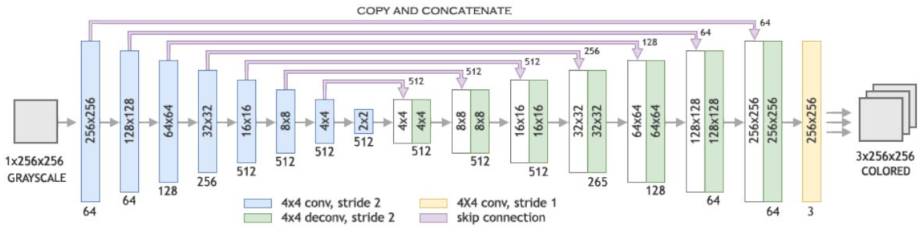
- Image Colorization - Technique of adding reasonable color information to monochrome images
- Difficult task as it involves translating a real-valued luminance image to a three-dimensional color value which does not have a unique solution.
- Has various application like - Image Restoration, Astronomical Photography, CCTV footage, electron microscopy etc.

Problem Statement

- Trained a Deep Convolutional GAN with UNET Generator to accomplish the task
- Studied the effects of Wasserstein Loss function along with its variants on model performance
- Used Places365 dataset for model training/testing.
- Peak Signal To Noise Ratio and Mean Absolute Error used as metrics to evaluate & compare performance of proposed models against baseline model

Deep Convolutional GANs

- GAN's fully connected layers replaced by convolutional layers consisting of upsampling instead of max-pooling
- Skip connections created to fix information bottleneck preventing the flow of low level information in the network (Feature Concatenation btw Symmetrically Opp. Layers)



- A min-max optimisation problem

$$\min_G \max_D V(G, D) = \mathbb{E}_x \left[\log D(x)\right] + \mathbb{E}_z \left[\log(1 - D(G(z)))\right]$$

Deep Convolutional GANs

Use of Binary Cross Entropy loss leads to problems:-

- ❖ Mode Collapse
  - ➢ Generator learns to consistently trick the discriminator defeating the entire purpose
  - ➢ Generator incapable of generating outputs belonging to various classes

- ❖ Vanishing Gradient
  - ➢ Gradients disappear when large input space mapped to a small one
  - ➢ Leads to zero learning in an iterative compounding scheme
  - ➢ Leads in a strong discriminator and weak generator

Wasserstein Loss

- Approximated the distance between real and generated distributions

$$\min_{g} \max_{c} \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

- Discriminator → Critic
- 1-Lipschitz Continuity (Norm of gradient ≤ 1) must be maintained
- Ensures Loss Function’s Continuity, Differentiability and Stability during training

Wasserstein Loss

To ensure 1-L continuity the following methods are used:-

- ❖ Weight Clipping
  - ➢ Forces the weight of critic to be constrained in an interval
  - ➢ Downside - Limits critic's ability to learn

- ❖ Gradient Penalty
  - ➢ Added as a regularisation term to W Loss ensuring continuity
  - ➢ Penalizes critic if gradient norm > 1
  - ➢ Done via sampling points by randomly interpolating between real and fake images $\epsilon x + (1-\epsilon)g(z)$

$$\min_{g}\max_{c}\mathbb{E}(c(x)) - \mathbb{E}(c(g(z))) + \lambda\mathbb{E}(||\nabla c(\hat{x})||_2 - 1)^2$$

Peak Signal To Noise Ratio

- PSNR is defined as the ratio between the maximum possible power of a signal and the power of undesirable losses that affect its representation.
- It is used to quantify reconstruction quality of images.

$$
\begin{aligned}
PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\
&= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\
&= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log
\end{aligned}
$$

Mean Absolute Error

- Mean of the absolute difference between the individual pixel values of the generated and original image.
- Represents closeness of generated image to original image

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}$$

Our Approach

1. **Train Base Model - DCGAN**
   Train standard DCGAN with UNet
2. **Train WGAN (Variant 1)**
   Replace BSE loss with W Loss, change optimizer & update training pattern
3. **Train WGAN + GP (Variant 2)**
   Add Gradient Penalty Term to W Loss, & remove Batch Normalisation
4. **Evaluate Metric on Testing Set**
   Tabulate results

Model Architecture DCGAN

- Generator and Discriminator use Convolutional Neural Network
- Skip connections in Generator between ith and (n-i)th layer
- N encoding and decoding layers
- Downsample and Upsample using 4*4 convolutional layer with stride of 2 followed by batch normalization
- Activation function: LeakyRelu (last layer of Generator uses tanh and last layer of discriminator use sigmoid)
- Optimizer: Adam Optimizer

Model Architecture WGAN

Same as DCGAN except for:-

- Loss Function - Wasserstein Loss
- Optimizer - RMSProp
- Critic weights clipped between [-0.01,0.01]
- Critic trained multiple times during each step but Generator trained only once

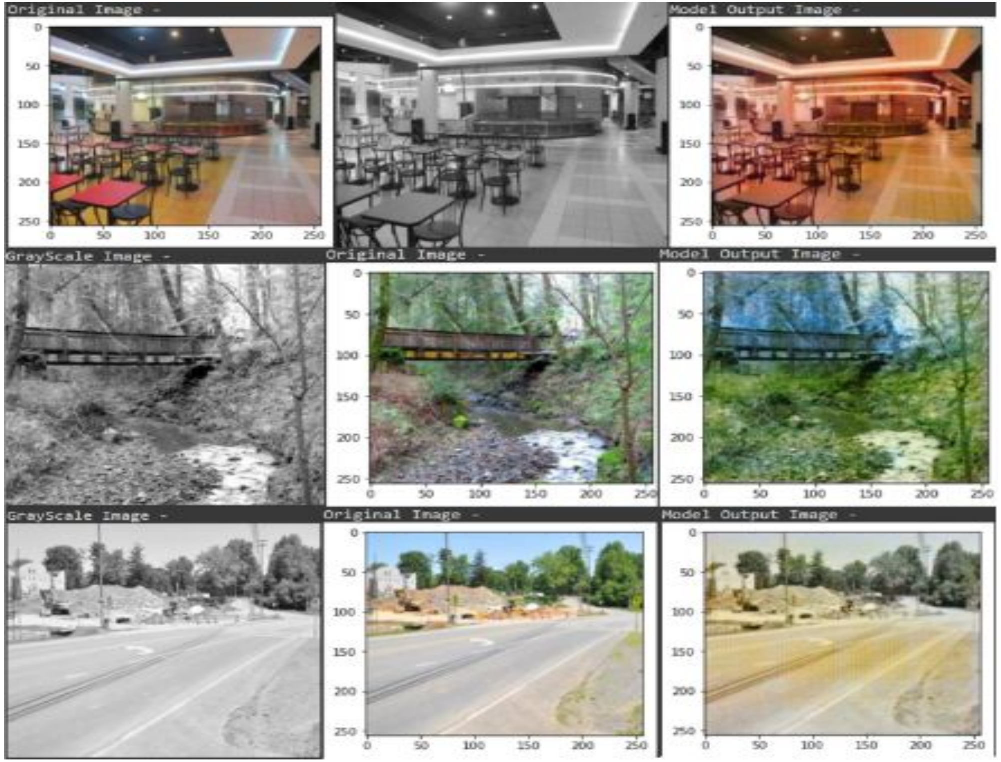
Model Architecture WGAN + GP

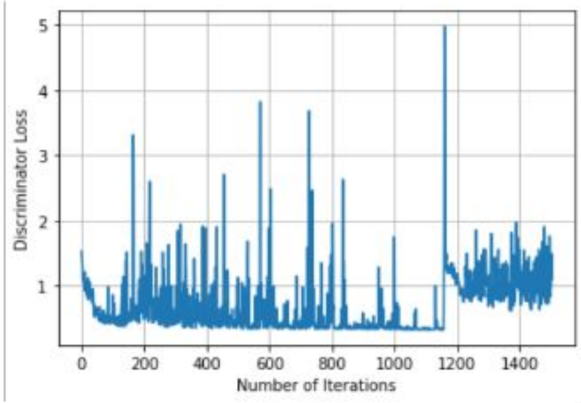
Same as WGAN except for:-

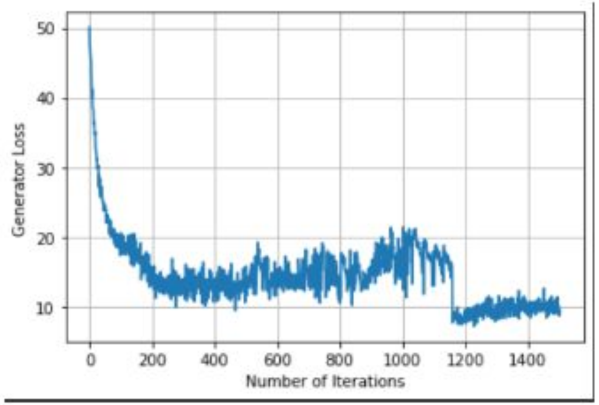
- Gradient Penalty term calculated and added to W Loss to ensure 1-L Continuity
- Remove Batch Normalisation from Critic

Note - All models were trained on 5000 images from Places365 dataset for 10 epochs with a batch size of 20

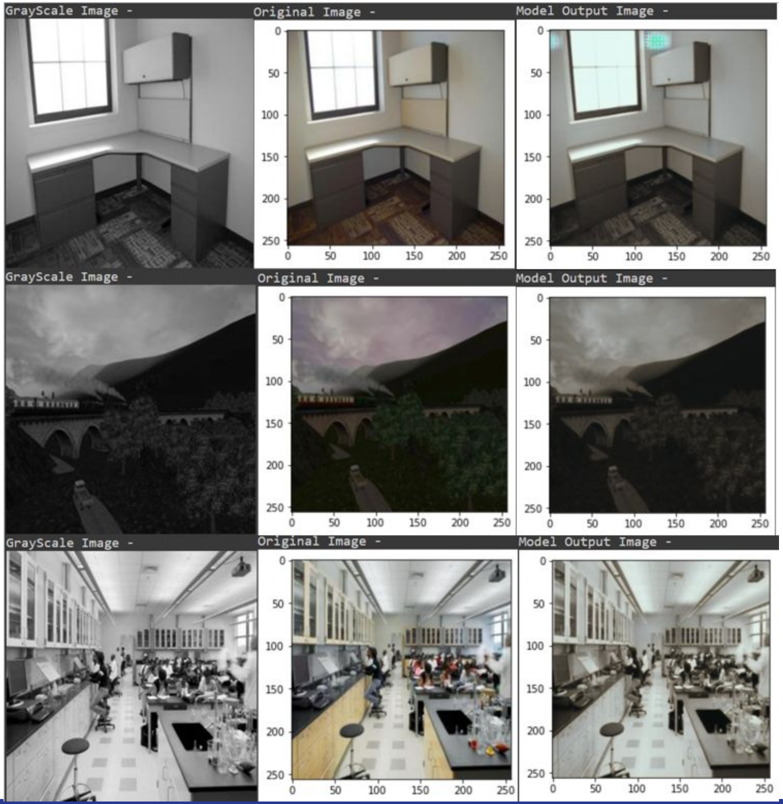
DCGAN Test Set Results:-

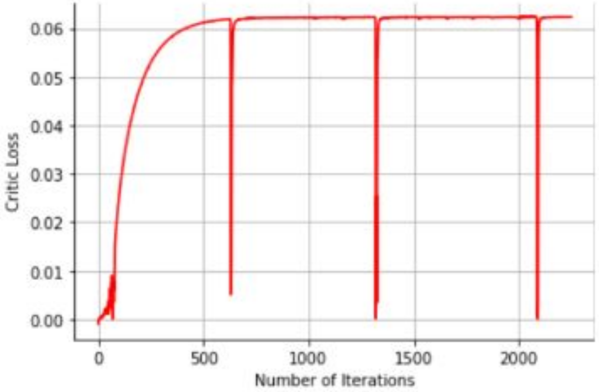


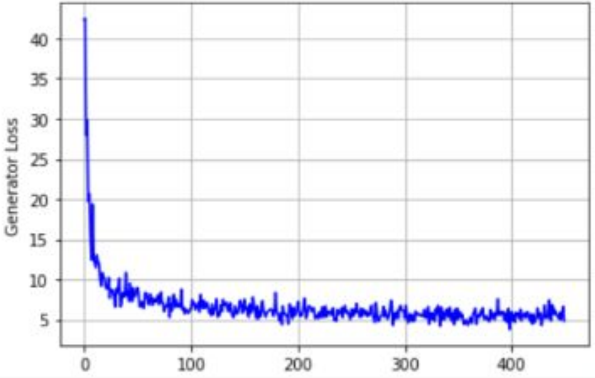




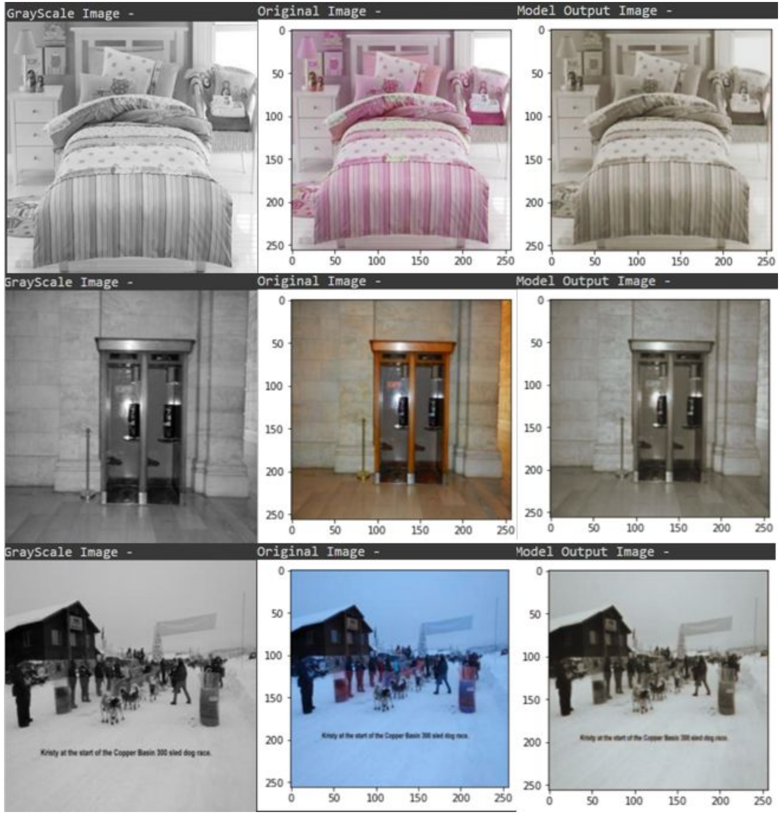
WGAN Test Set Results:-

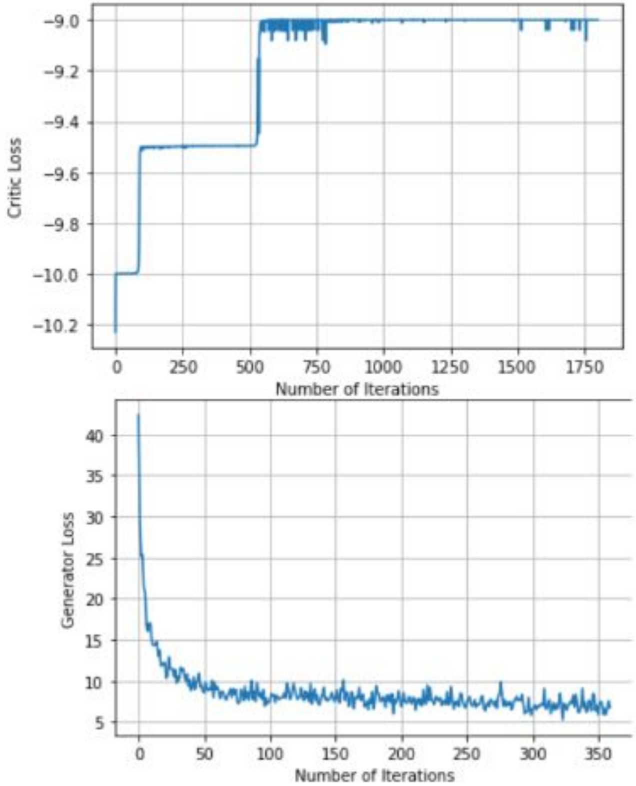






WGAN + GP Test Set Results:-





Results

- An increase in PSNR indicates better quality of generated image
- An decrease in MAE indicates that generated image is closer to original colored image

Model	PSNR	MAE
Vanilla DCGAN	6.25	113.34
WGAN	6.27	111.46
WGAN with Gradient Penalty	6.29	110.69

Conclusion

- PSNR Metric increases in the following order DCGAN → WGAN → WGAN + GP which indicates that the quality of the output images from the generator has improved

- MAE Metric decrease in the following order DCGAN → WGAN → WGAN + GP which indicates that the generated image is closer to the original colored image

- Good quality images were not observed due to insufficient training data and epoch number due to GPU limitations.

Thank You