Proceedings of 12th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2022 Month: October Volume: 12

Auto Colorization of Gray Images using GAN

Prabin Nepali [a], Rupesh Kumar Sah [b], Surendra KC [c]

[a, b, c] *Paschimanchal Campus, IOE, Tribhuvan University, Nepal*
[a] prabino.nt@gamil.com, [b] rupesh@wrc.edu.np, [c] kcsurendra28@gmail.com

Abstract
Colorization of grayscale images was done from decades ago. However, the methodologies has upgraded from coloring images manually working in Photoshop (photo editing tool) to working with machine learning algorithms and further advanced to deep learning methods. This research introduces a simple technique of converting grayscale images into color images with the help of Generative Adversarial Network (GAN). Basically, it introduces a technique for predicting color of a gray image with the use of U-net architecture in the generator module while PatchGan in the discriminator module monitors the generated images from generator as fake or real. This method provides a comprehensive iterative method for color transfer mapping from gray to chromatic values of the image with continuous learning from generator and discriminator loss function. The LAB image format is taken into account to lessen the prediction probabilities of color channels for grayscale images. Additionally, special consideration is taken into account to train GAN using the pre-trained model ResNet-18 that enhances the evaluation of colorless images so that only chromatic information is transferred and the target image's original brightness values are retained. FID values also shows the model is comparatively good.

Keywords
Photoshop, U-net, PatchGAN, LAB, ResNet-18, Gray scale image, FID

1. Introduction

Auto colorization of grayscale image is one of the popular Image to Image translation application that is widely researched and experimented with different machine learning, and deep neural learning network techniques. Because early ancient and historical pictures are depicted in grayscale, colorization can disclose additional color information while preserving many of the aesthetic and perceptual aspects. [1]

Grayscale Image: An image that simply contains luminous intensity information is called a grayscale image. These kinds of images that range from black with the weakest intensity to white with the maximum intensity, are also referred to as monochromatic images. Between the black and white color spectrum, grayscale photographs have several shades of gray.[2] The 8-bit color format, with a color gamut of 0-255, is another name for it. 127 denotes gray color, 255 denotes white, and 0 denotes black. The equation according to Weighted method or Luminosity method for gray image is set as: New grayscale image = ((0.3 * R) + (0.59 * G) + (0.11 * B)) where, Red color contributes 30%, Green contributes 59% which is the greatest in all three colors and Blue contributes the least 11%.

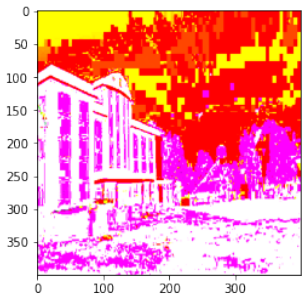

Figure 1: MSC Bulding WRC in LAB format.

Lab images format: As per Patil, Akshay, et al. [2] research Lab color space is image format where L abbreviates for Luminance and a, and b have two color combination meaning. In addition 'L' channel models luminance which is a grayscale image thus the model has to predict values for two channels (a*b), and after its prediction L channel can be concatenated to get the colorful image. Since Euclidean distance more closely resembles how people perceive color differences in

Lab, Lab is chosen over RGB.[3]. In an 8-bit unsigned integer image, the prediction of 3 color pixels for RGB would be 256^3 while in LAB the prediction of two channels a and b is 256^2. Thus using L*a*b aids the performance and the efficiency of any model. The paper [1] [3] [4] have used the CIELAB format as well.

In addition, the LAB version of the image has three important pieces of information. Luminance informs about light intensity (the darkness or lightness) which varies between 0 to 100. 0 implies the dark intensity of a pixel while 100 implies the maximum brightness of the image pixel. The letters "A" and "B" imply that the color balance between green-magenta, and blue-yellow respectively. These vary between -128 and 127. [5].Figure 1 represents the LAB format while Figure 2 illustrates each L, A, and B channel.

Figure 2: All three channel of LAB.

The first section of Figure 2 holds L – luminous intensity information which is a grayscale image of the MSc building of Western Regional College and 'a – green - magenta channel which is the second section of the figure which shows how green or magenta the image is and the last section 'b' blue-yellow informs about blue-yellow channel intensity of the image.

RGB images:Color images are often generated with combination of several combination of primary and secomdary color channels, each of them representing value levels of the given channel. Red, green, and blue are the three distinct channels that make up an RGB picture. It features a 24 bit color format, which divides the three channels into eight bits each, and a 16 bit color format, which uses five bits for Red, six bits for Green, and five bits for Blue.

The Figure 3 displays MSC building of WRC,Pokhara in all three R,G and B channel.

1.1 Generative Adversarial Networks (GANs)

The discriminator D and the generator G networks, which make up GANs; the main goal is to compete with one another to produce synthetic realistic images

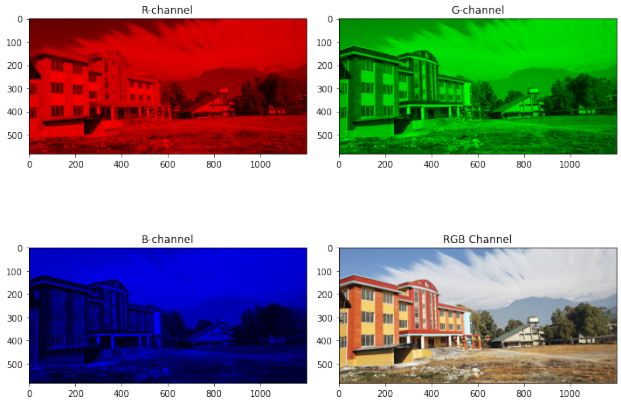

Figure 3: All three channel of RGB.

and, as a consequence, learn functional transformations that go from noise distribution to data distribution. While the discriminator D tries to distinguish between the two, the generator G learns to trick it by producing almost real data distribution pdata from the dataset. Usually Generator G begins its learning process with a latent noise z drawn from a normal distribution N(0,1), whereas Discriminator D begins its learning process by estimating the distributions of actual and produced data.[6] [7]. However, Grayscale photographs, rather than noise, are used as inputs to the automated colorization in the research. The generator's input is viewed as zero noise since no noise is introduced, [5]. The figure 4 illustrates the basic architecture of GANs:

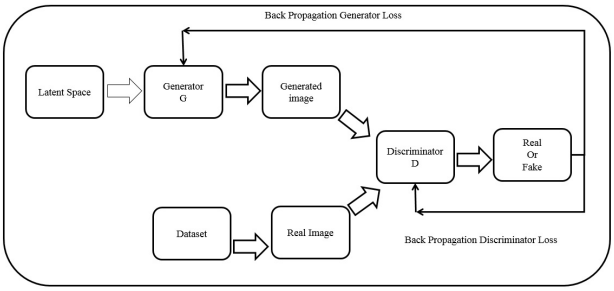

Figure 4: Basic Architecture of GAN model.

In order to train GANs, one must first predict the discriminator parameters that will most accurately classify image, and then one must choose the generator parameters that will most effectively confuse the discriminator with its output. The objective function V(G,D), which depends on both the generator and the discriminator, is used to determine the training cost.

GAN's training objective function can be illustrated as

follows:

$$V(\min_G, \max_D) = \mathbb{E}_{pdata(x)}[\log D_(x)] + \mathbb{E}_{pg(x)}[\log[1 - D_{(G(z))}]] \quad (1)$$

where pdata(x) is Real sample, pg(x) is input noise, and G(z) is a generated sample. The Generator network G is trying to lessen the loss function against an adversary Discriminator network which tries to maximize the loss function.

During the optimization process, the generator and the discriminator losses frequently fluctuate without converging to a optimal stopping point, causing it to enter oscillation mode and yield no results. It is difficult to determine when the GAN has ceased training because to the lack of a solid stopping criteria. Furthermore, mode collapse occurs when a GAN's generator becomes caught in a loop, repeatedly creating one of a few types of samples. The methods must discover a straight mapping from grayscale to color picture space. The generator model employs a U-net with skip connections in order to avoid information bottleneck that limits the flow of low-level information in the network.

All colorization algorithms aim to replace the scalar value of luminous intensity contained in each pixel of a grayscale picture with two-dimensional color space vectors 'a' and 'b' and display the LAB format into RGB color format.Images can be better represented into color images with use of GAN models and its variants [3]. Colorization of photographs is a difficult subject owing to the variety of imaging situations that must be handled by a single algorithm. To address the issue, picture colorization models ranging from simple machine learning methods to Deep neural networks have been presented. The time and environment can be deduced properly. A smart system which can literally observe the surroundings and assign a color to the pixel is required. Thus this proposed model can state the problem and avoid hard-coding and Photoshop edits.

The main objective of the proposed research is to develop a methodology based on Image-to-Image translation (I2I) to generate RGB color images from gray image with the use of GAN model using U-Net as generative module to generate the synthetic RGB images in coordination with PatchGAN that is used in the discriminative module to verify if the generated image is close to real image.

2. Related Works

Different approaches are seen to colorize grayscale images in recent years advancing from Machine learning to CNN codes to Deep Neural Network.

In 2022, B.K. Nyaupane et al. [1] proposed Grayscale Image Colorization using MSE, BCE and MAE losses with use of U-Net in generator and PatchGAN in discriminator. The input dataset was self assembled dataset which is a subset of Places365. The self-built dataset contained RGB 5600 color images of 128x128 dimension are used. 5000 images are used for training and remaining 600 images are allocated for testing. Moreover, it is comparison of BCE loss in discriminator and MSE and MAE loss in generator in first model, whereas second model consists MSE loss in discriminator and BCE and MAE loss in generator. Loss score of the first model is stable whereas color image from second model seemed better.

For grayscale picture colorization, Q.Fu et al. [8] implanted ConvNet and Generative Adversial Networks (GAN) architectures in 2017. ConvNet is built as an encoder-decoder to provide output with pixel-wise "L2" loss function. Models are trained using animated visuals from the popular video game series "Pokemon."

K. Nazeri et al. [5] created a model for grayscale picture colorization in 2018 utilizing U-net as the baseline model and a conditional Deep Convolutional Generative Adversarial Network (DCGAN). Both models are trained on the CIFAR-10 and Places365 datasets.This research also used LAB format for image processing.

Anwar, Saeed et al. [9]present a comprehensive evaluation of recent state-of-the-art deep learning-based picture colorization algorithms, including explanations of their main block structures, input, optimizers, loss functions, training procedures, and training data, among other things. This study categorizes existing colorization techniques into seven groups and investigates critical performance factors such as benchmark datasets and assessment criteria. It also highlights the shortcomings of current datasets and introduces a new colorization-specific dataset. The Natural-Color Dataset, a new benchmark dataset, is presented in the paper (NCD).

3. Methodology

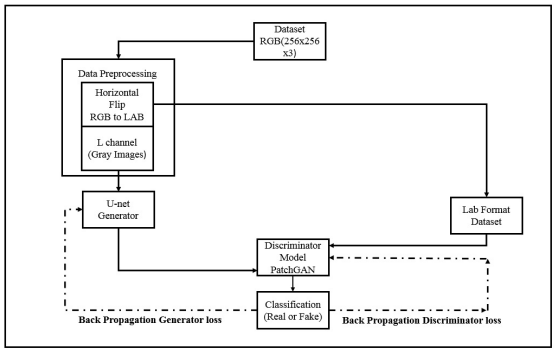

Figure 5: Block diagram of the Model.

The figure 5 illustrates the overall topology of the proposed methodology. The input to the proposed model is a RGB images which is taken from Grayscale2Colorization_Mixed _Dataset [1] a subset of Places365 that contains 10,000 images each of size 128x128. In the data processing block the RGB images are first converted to L*a*b format where L channel holds only the Luminance intensity and can be taken as grayscale image. The gray scale images are then given to generator to generate their RGB images.

3.1 U-net (The Generator)

The U-Net design is based on Long, Shelhamer, and Darrell's fully convolutional network. U-Net's design is similar to that of an auto-encoder network. The main distinction is that U-net features skip connections between the layers in the encoder and decoder components of the generator, whereas auto-encoder does not.

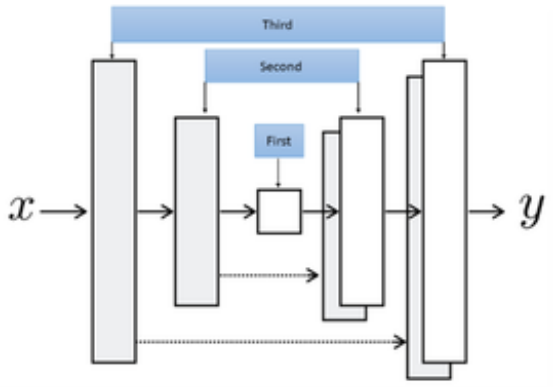

Figure 6: Basic U-Net structure.

The figure 6 illustrates the architecture of U-net.The U-Net is divided into three sections. Encoder, Bridge, and Decoder.The encoding component is in charge of transforming the input picture into a compact representation known as the latent space of the input. The output of the first layer is combined with the output of the last layer, as shown in the picture. The output of the second layer is combined with the output of the second last layer, and so on.

The encoder phase of the proposed model contracts the input pictures with down sampling operations. The model input x is a grayscale picture with dimensions of 256x256x1, and the center region of the U-net receives a 1x1x512 latent vector, as shown in Figure 7 The contraction section is made up of eight convolutional layers with 4x4 convolutions and a stride of 2. In the encoder, a Leaky ReLu activation of 0.2 is applied after each convolution layer.

The decoding procedure comprises the reconstruction of the input picture, which is supplemented by successive layers of a conventional contracting network, where upsampling operators substitute pooling operations. As a result, these layers boost the output resolution to 256x256, which is a created LAB format picture.

In the proposed model, the expansion takes place in decoder with the use of upsampling operations which consists of the feature map (2x2 up convolution). A 4x4 convolution is performed after each upsampling and appended with the corresponding feature map taken from the encoder that helps to reconstruct the information with accuracy.

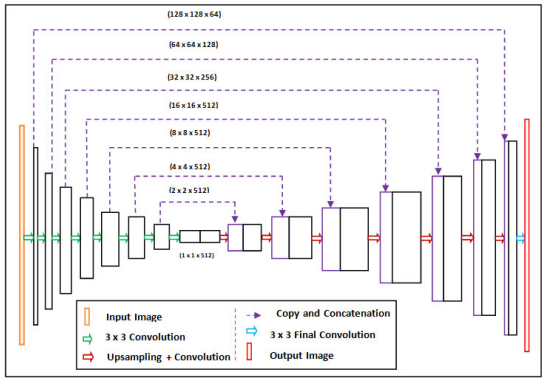

Figure 7: UNet Generator concatenation.

The presence of a large number of feature channels in the upsampling section of U-Net allows the network to transport context information to higher resolution

layers. As a result, the expanding process is almost symmetrical to the contracting process, resulting in a U-shaped architecture as seen in figure 7. In U-net architecture the downsampling and upsampling networks are linked by a bottleneck, and the skip connections replicate and concatenate feature maps from the encoder to decoder.Without any completely linked layers, the network merely uses the valid part of each convolution. By mirroring the input image and forecasting the pixels in the border region of the image, the missing context is inferred. This method is ideal for high-resolution photos that demand a lot of GPU memory.

3.2 PatchGAN (Discriminator)

A PatchGAN is a convolution network with discriminator receptive fields of 70x70 patches in the input picture. The discriminative convolution layer is depicted in Figure 8. The created picture of size 256x256x3 is patched on size 70x70x3 as the generator's output. The patch is processed using 3x3 convolution with maximum pooling. Before the activation function, instance normalization is employed, and LeakyReLu is applied to each layer. The discriminator is a PatchGAN that penalizes structure solely at the scale of local image patches. It tries to determine whether each NxN patch in a picture is authentic or fraudulent. As a result, the input picture for the discriminator is first separated into patches of square length N. The discriminator patch is applied convolutionally to the input picture, and the final output is produced by averaging the responses from all of the patches. The PatchGAN discriminator essentially models the picture as a Markov random field, with pixels separated by more than a patch diameter assumed to be independent. The reason for utilizing PatchGan is because, despite the fact that the patch N is smaller than the entire picture, it has less parameters and runs faster for any image dimension while still producing high-quality results.[10]

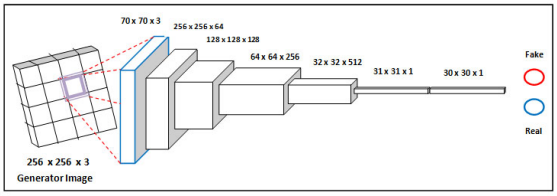

Figure 8: Convolution layer in PatchGAN.

The discriminator block has the access to both the real RGB images and the generated RGB images. Thus it compares the loss function which is then given to the generator block as feedback by which generator block learns further to generate the better RGB images. The better generated RGB images are again compared at discriminator block, this loops runs until discriminator loss stabilizes. By then the Generator block is capable to fool the discriminator block.

3.3 ResNEt-18

However, due to inefficiency of the proposed model, a small change was made to the model. The generator end was trained using pre-trained model ResNet-18 separately and the whole model was concatenated at the end.

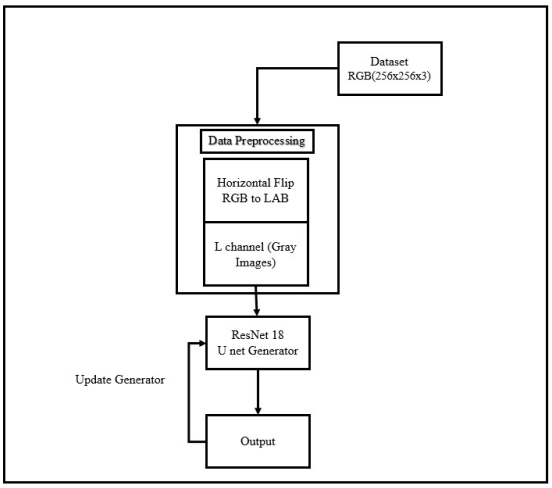

Figure 9: Block diagram using ResNet-18.

The figure 9 illustrates how a generator end is replaced with the pre-trained model and later on appended to the entire model of Figure 5 with an expectation of better results. The results were promising after the amendment in the model which is illustrated in Section 5 Experimental Result and Output where the reconstructed images Figure 14 looks better than the earlier iteration without ResNet-18 in figure 12 which has more of blue and hue tone in it. ResNet-18 is a convolutional neural network made up of 18 layers. It is a pretrained version of the network that has been trained using over a million photos from the ImageNet collection. It can identify photos into 1000 different item categories. As a result, the network has acquired rich feature representations for a broader domain of pictures, making this a useful tool for any model.

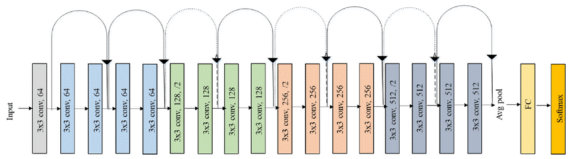

Figure 10: Architecture of REsNet-18.

The figure 10 is the basic architecture of the ResNet-18. The [3] shows classification and segmentation increased 5% and 2% respectively. Thus the ResNet-18 pre-trained model is used in the proposed model to train Generator before passing with the presented dataset. Then it is further fine-tuned for the proposed task. [3] [11] This method provides a good starting point since the generator already comes with learned basic features.

3.4 Loss Functions

GAN training entails both determining the parameters of a discriminator that maximize its classification accuracy and determining the characteristics of a generator that confuse the discriminator the most. The cost of training is calculated using a value function, V(G,D), which is affected by both the generator and the discriminator [5]. This is also known as Minimax GAN Loss.

$$V(\min_G \max_D) VLSGAN(D,G) = \mathbb{E}_{pdata(x)}[\log D(x)] + \mathbb{E}_{pg(x)}[\log[1 - D_{(G(z))}]] \quad (2)$$

where pdata(x) is Real sample, pg(x) is input noise, and G(z) is a generated sample.

G(z) is a generated sample. This loss is also defined as binary cross entropy.

However, Least Squares Generative Adversarial Network (LSGAN) is also used; the loss function penalizes the generated images based on the distance measured from decision boundary. The objective function is defined as follows:

$$\min_D VLSGAN(D,G) = \frac{1}{2}\mathbb{E}_{pdata(x)}[D(x) - b]^2 + \frac{1}{2}\mathbb{E}_{p^z}[D_{(G(z)-a)}]^2 \quad (3)$$

$$\min_G VLSGAN(D,G) = \frac{1}{2}\mathbb{E}_{pdata(x)}[D(x) - c]^2 \quad (4)$$

where a and b are the labels for the fake data and real data represents the value that G prefers D to believe for fake data.

The above listed BCE or MSE loss trains the discriminator model. The Mean Average Error MAE loss or L1 loss is used to calculate the generator loss by finding the difference between the generated image of the source image and the expected target image. Thus the new loss function which is the combination of discriminator loss function MSE or BCE and generator loss function MAE; called composite loss function is calculated.

Finally $G_{Loss} = BCE\ or\ MSE + \lambda MAE_{Loss}$

where λ is a new hyperparameter which controls the composite loss here in the proposed model it is set as 100 i.e. the MAE loss is 100 times the important than the BCE or MSE loss for generator during the learning process.

The Adam optimizer and weight initialization described by [5] [10] [7] are used to train the network, with a learning rate of $2x10^{-4}$ for both the generator and discriminator models. The hyper-parameter λ is set to 100, forcing the generator to create pictures that are similar to the ground truth.

4. Dataset Description

Following data set are used for the current research work. The details about the data set are given below: The Grayscale2Colorization_Mixed_Dataset is a subset of the Places365 dataset used in [1]. The Places365 dataset is a collection of scene images from different places which is composed of 10 million images comprising 434 scene classes with two versions of the dataset; Places365-Standard and Places365-Challenge2016. The Grayscale2Colorization_Mixed_Dataset is a ready-made dataset assembled from Places365 dataset. It consists up of total 10,000 train images of size 128x128 and the test dataset contains 457 images.

Due to hardware inefficiency, out of the images 10,000 images are selected for the proposed model. The proposed model used 8,000 images for training purpose while 2,000 images are used for validation. The images are further converted to 256x256 dimensions.

4.1 Data pre-processing

Data pre-processing block converts RGB images into L*a*b format and resizes the images to 256x256. In a 8 bit unsigned integer image the prediction of 3 color pixel for RGB would be 256^3 while in LAB the predicting two channels a and b is 256^2. Thus using L*a*b aids performance efficiency of any model. Further, L channel of L*a*b image format gives gray image which is input to the generator. The model has to predict simply value of ‘a’ and ‘b’.

5. Results and Discussion

5.1 Experimental and Implementation Details

The programming language used in this research is Pytorch. Developed largely by Meta AI, PyTorch is an open source machine learning framework based on the Torch library used for a variety of computer applications, including computer vision and natural language processing.

GPU and TPU of free and paid version are available without initial set ups. Google Colaboratory is a free / paid version cloud-based Python environment;hosted on Jupyter notebook which have powerful computing potentialities where researches may run codes. For the proposed model the Google Colaboratory Pro is used for the training purpose.

5.2 Results and Discussion

Various attempts and trials have been made to come up with the color images from the grayscale images as entitled in this proposed model. All the attempts have been made on the Google Colaboratory.

The training details and obtained results are:

Attempt: 1

Dataset: Grayscale2Colorization_Mixed_Dataset
Total images: 10000
Training images: 8000
Testing Images: 2000
Hyperparameter: Adam 0.5
Image Batch size: 100
Generator and Discriminator Learning Rate: 2×10^{-4}
Activation function: LeakyRelu 0.2
Iteration: 5500

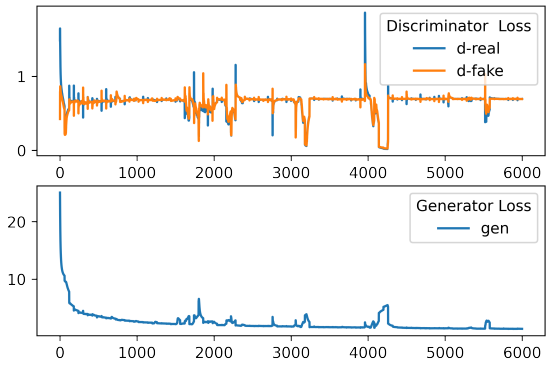

Figure 11: Loss plot before ResNet

Figure 12: Images Plot before ResNet

In the figure 11 the first section shows Discriminator loss of fake and real images while second section gives the loss plot of generator before addition of pre-trained model ResNet-18.

According to figure 12 the first section shows grayscale image of the input image while second section gives the output of the model and the third section is ground truth

Attempt: 2
Dataset: Grayscale2Colorization_Mixed_Dataset
Total images: 10000
Training images: 8000
Testing Images: 2000
Hyperparameter: Adam 0.5
Image Batch size: 16
Generator and Discriminator Learning Rate: 2×10^{-4}
Activation function: LeakyRelu 0.2
Iteration: 160000

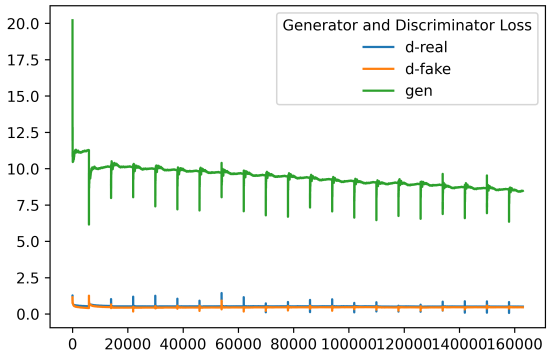

Figure 13: Generator and Discriminator Loss

As in the research work of [3] [9] and many other super resolution journals pre-trained models have been used in order to come up with better results. Thus the proposed model was changed slightly in the generator section using ResNet-18 which has the 18 layers. The ResNet-18 was used for the training the model without connecting the discriminator part.

Figure 14: Image Plot After ResNet

It is observed in figure 13 that the generator loss starts high and spikes certain interval then stabilizes after 20000 iteration in around 8. The discriminator loss keeps track of fake and real images loss with real images plot on blue color and fake images loss with orange color which is also stabilizing in around 0.3.

The generated images from generator is relatively better than before attempts as in the figure 14

5.3 Evaluaion Metrics

The performance of a model is elaborated by the evaluation metrics. It shows if a model is heading towards desirable output with maximum accuracy.

Structural Similarity SSIM is a tool for calculating how similar two photos are to one another. Its value varies from 0 to 1. Higher the value more similar are the images compared.

A popular measure to evaluate the picture quality produced by a generative adversarial network, is the Fréchet inception distance (FID). FID evaluates the distribution of produced pictures against that of authentic images. FID score ranges from 0 to 1. Lower FID values relays the model is good.

The MSE calculates the average square of the errors, or the average square of the discrepancy between the estimated and actual values.

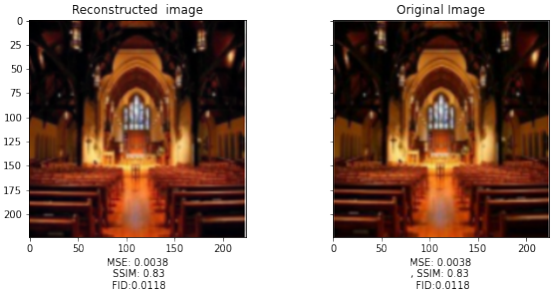

Figure 15: Evaluation Metrics

The Figure 15 shows the three above evaluation metrics with their values. The MSE values 0.038, SSIM value reads 0.83 and FID score is 0.0118.

5.4 Output

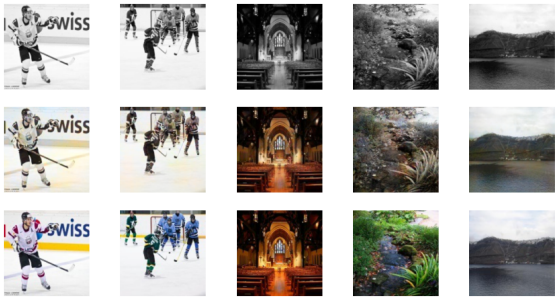

Figure 16: Final Output

With the above presented model and its hyper parameter, 4 pictures were tested and the final output is illustrated in Figure 16. The first section shows grayscale image of the input image while second section gives the output of the model and the third section is ground truth. Figure 16 is testing phase results.

6. Conclusion

The model achieves its goal of coloring grayscale photos. The approach overcomes the issue that hand coloring software have. According to the results, the generated photographs deviate somewhat from the actual images. However, the evaluation phase has yet to be completed. The pixel deviation may be lowered even more with more photographs for training and upgraded processing units.

7. Limitations

Mis-colorization was a common issue in photographs with a high level of textural detail because the dataset comprises more photographs of sky and grass, the research generalizes sky, grassland, and soil colors. Thus unfamiliar objects like hats, cars, balls were sometime colored inappropriately when comparing to ground truth. Many of the generated images from U-net had a brown and hue tone which is commonly known as the “Sepia Effect” usually found in the LAB color format as discussed in [5] Nazeri et al. Thus ResNet-18 was implemented in order to subside the issue. These issue might overcome with increased

number of dataset and training with unlimited TPUs and storage.

References

[1] Bal Krishna Nyaupane, Rupesh Kumar Sah, Bishnu Hari Paudel, and Subarna Shakya. International journal of advanced engineering.

[2] Akshay Patil, Asmit Save, Varun Patil, and Vanessa Dsouza. Coloring greyscale images using deep learning. 2019.

[3] Sandra Treneska, Eftim Zdravevski, Ivan Miguel Pires, Petre Lameski, and Sonja Gievska. Gan-based image colorization for self-supervised visual feature learning. *Sensors*, 22(4):1599, 2022.

[4] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[5] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In *International conference on articulated motion and deformable objects*, pages 85–94. Springer, 2018.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[7] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[8] Qiwen Fu, Wei-Ting Hsu, and Mu-Heng Yang. Colorization using convnet and gan. *in Stanford University*, pages 1–8, 2017.

[9] Saeed Anwar, Muhammad Tahir, Chongyi Li, Ajmal Mian, Fahad Shahbaz Khan, and Abdul Wahab Muzaffar. Image colorization: A survey and dataset. *arXiv preprint arXiv:2008.10774*, 2020.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. pages 1125–1134, 2017.

[11] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593. Springer, 2016.