TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Khwopa College Of Engineering
Libali, Bhaktapur
Department of Computer Engineering

A PROGRESS REPORT
ON
IMAGE COLORIZATION USING GAN

Submitted in partial fulfillment of the requirements for the degree

BACHELOR IN COMPUTER ENGINEERING

Submitted by

Rabit Khaitu	KCE076BCT028
Romik Gosai	KCE076BCT032
Sagar Suwal	KCE076BCT034
Saman Chauguthi	KCE076BCT035

Under the Supervision of
Er. Niranjan Bekoju
Department of Computer Engineering

Khwopa College Of Engineering
Libali, Bhaktapur
2023-24

Certificate of Approval

The undersigned certify that the final year project entitled **"Image Colorization using GAN"** submitted by Rabit Khaitu, Romik Gosai, Sagar Suwal, Saman Chauguthi to the Department of Computer Engineering in partial fulfillment of requirement for the degree of Bachelor of Engineering in Computer Engineering. The project was carried out under special supervision and within the time frame prescribed by the syllabus.
We found the students to be hardworking, skilled, bona fide and ready to undertake any commercial and industrial work related to their field of study and hence we recommend the award of Bachelor of Computer Engineering degree.

.....
Er. Niranjan Bekoju
Project Supervisor
Machine Learning Engineer
Fuse Machines Nepal Pvt. Ltd.

.....
Er. Dinesh Gothe
Head of Department
Department of Computer Engineering,
Khwopa College of Engineering

Acknowledgement

We would like to express our deep gratitude to Er. Dinesh Man Gothe for his invaluable guidance, unwavering encouragement, and steadfast support that have been instrumental in the successful completion of this report. His insightful advice and mentorship have truly paved the way for our project's development. We would also like to express our sincere appreciation to all the teaches for there invaluable assistance and support throughout this endeavor.

We are also immensely thankful to our dedicated supervisor Er. Niranjan Bekoju, whose expertise and insights have provided us with a solid foundation and direction. His continuous guidance and constructive feedback have been instrumental in refining our project's scope and approach.

Furthermore, we wish to extend our thanks to our fellow classmates, whose camaraderie and discussions have provided us with diverse perspectives and inputs that have enriched the quality of this report. Their indirect contributions have undeniably shaped the project's trajectory.

Lastly, we wholeheartedly acknowledge the countless individuals who have played both direct and indirect roles in the realization of this proposal. Their collective efforts have created an environment conducive to the pursuit of knowledge and innovation. It is with profound appreciation that we acknowledge all those who have been part of this journey, contributing to its successful fruition

Rabit Khaitu	KCE076BCT028
Romik Gosai	KCE076BCT032
Sagar Suwal	KCE076BCT034
Saman Chauguthi	KCE076BCT035

Abstract

The project focuses on colorizing grayscale images of Nepalese monuments using Pix2Pix Generative Adversarial Networks (GANs), with a specific emphasis on RGB to YUV conversion for enhanced color representation. The dataset, exclusively curated for this study, pairs grayscale representations of monuments with their colored counterparts. The Pix2Pix model, comprising a U-Net generator and discriminator, undergoes specialized training to preserve historical landmark characteristics. The study details dataset preparation, model architecture, and training specifics, demonstrating the application of Pix2Pix GAN and RGB to YUV conversion to automate and enhance the visual representation of Nepal's cultural heritage. Evaluation involves testing the model on monument-specific images, showcasing its adaptability and efficacy in this domain. The outcomes contribute to advancing image colorization techniques and play a significant role in preserving Nepal's rich cultural heritage through deep learning technologies and improved color accuracy with RGB to YUV conversion.

Keywords: *Image Colorization, Computer Vision, Generative Adversarial Network, pix2pix*

Contents

List of Tables

List of Figures

List of Symbols and Abbreviation

AI	Artificial Intelligence
API	Application Programming Interface
CIFAR	Canadian Institute For Advanced Research
CNN	Convolutional Neural Network
CSS	Cascading Style Sheets
DCGAN	Conditional Deep Convolutional Generative Adversarial Network
GAN	Generative Adversarial Network
HTML	Hypertext Markup Language
MC-GAN	Multidomain Cycle-consistency Generative Adversarial Network
MSE	Mean Squared Error
OpenCV	Open Source Computer Vision Library
PSNR	(Peak Signal-to-Noise Ratio
SAR	Synthetic Aperture Radar
SCGAN	Saliency Map-guided Colorization with Generative Adversarial Network
SSIM	Structural Similarity Index
VAE	Variational Autoencoder
VGG	Visual Geometry Group
VSCode	Virtual Studio Code
UI	User Interface

Chapter 1

Introduction

1.1 Background

Colorization is the process of adding plausible color information to monochrome photographs or videos [1]. Currently, digital colorization of black and white visual data is a crucial task in areas so diverse as advertising and film industries, photography technologies or artist assistance [2]. As colorization requires estimating missing color channels from only one grayscale value, it is inherently an ill-posed problem. Moreover, the plausible solutions of colorization are not unique (e.g., cars in black, blue, or red are all feasible results). Due to the uncertainty and diversity nature of colorization, it remains a challenging task [3].

There are some earliest conventional image colorization methods which can be divided into color transfer-based method (or example-based method) [4] [5] and scribble-based method [6]. In method proposed by Welsh *et al.* [4], colorization was done by matching luminance and texture information between an existing color image and the grayscale image to be colorized. However, this proposed algorithm was defined as a forward problem, thus all solutions were deterministic. The color transfer-based image colorization method proposed by Reinhard *et al.* [5]. The basic idea of this kind of method is to select a color image as reference image, and the color information of the reference image is transmitted into the target gray image that is the high similarity to the reference images. The generalization ability of the transfer-based method is not well, because the process requires a highly similar reference image with the source image. In 2012, Gauge *et al.* [7] proposed an example-based colorization method; this method first analyzed a set of sample color images and extracted the coherent regions of homogeneous textures, and then the grayscale image can be colored by the similar texture of a color image.

In 2014, Goodfellow *et al.* [8] proposed a new type of generative model: generative adversarial networks (GANs). A GAN is composed of two smaller networks called the generator and discriminator. As the name suggests, the generator's task is to produce results that are indistinguishable from real data. The discriminator's task is to classify whether a sample came from the generator's model distribution or the original data distribution. Both of these subnetworks are trained simultaneously until the generator is able to consistently produce results that the discriminator cannot classify.

In [9], a colorization method was proposed by comparing colorization differences between those generated by convolutional neural networks and GAN. The mod-

els in the study not only learn the mapping from input to output image, but also learn a loss function to train this mapping. Their approach was effective in ill-posed problems such as synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images. We aim to extend their approach by generalizing the colorization procedure to high resolution images and suggest training strategies that speed up the process and greatly stabilize it.

1.2 Problem Statement

The Image Colorization project aims to address the conversion of grayscale or black-and-white images into colorized versions that closely resemble the original colored counterparts. By adding realistic and accurate colors to these images, the project seeks to enhance their visual appeal and provide a more immersive experience for viewers. This is particularly important in the context of historical preservation, where colorizing images can bridge the gap between the past and the present, enabling a deeper understanding and appreciation of historical events, places, and people. Additionally, colorization enhances visual quality, conveys important contextual information, promotes accessibility for color-blind individuals, allows for artistic expression, facilitates interactive applications, and serves as a powerful tool for education and documentation. By addressing the challenges associated with image colorization, the project aims to unlock the full potential of grayscale images, enrich visual experiences, preserve historical accuracy, foster understanding and creativity, and have broader implications across various industries.
Even though various Image colorization models are built for different domains, there are very few work done on the domain of traditional monuments of our country. We aim to focus our work on colorizing old monuments photograph whose existence is not available today or there are various modifications done to those monuments and their surroundings.

1.3 Objective

The main aim of this project is:

- To develop a platform that enables conversion of grayscale images of old monuments photographs into colorized version.

Chapter 2

Literature Review

S. Wan, Y. Xia, L. Qi, Y.-H. Yang, and M. Atiquzzaman (2020) [10] introduces an automatic image colorization method using neural networks and optimization. By segmenting grayscale images, extracting features, and propagating color points to neighboring pixels, the proposed method achieves superior colorization results compared to existing algorithms, as demonstrated in experiments on various images.

A. Deshpande, J. Lu, M.-C. Yeh, M. J. Chong, and D. Forsyth (2017) [11] introduces a method for generating multiple diverse colorizations for grayscale images. They use a variational autoencoder (VAE) to learn color field embeddings, incorporate loss terms to prevent blurriness and handle color distribution, and show that their approach outperforms other models in producing diverse colorizations.

B. Li, F. Zhao, Z. Su, X. Liang, Y.-K. Lai, and P. L. Rosin (2017) [12] presents a novel example-based image colorization method that utilizes a sparse representation at the superpixel level. By incorporating various features and a dictionary-based sparse reconstruction approach, the proposed method automatically colorizes grayscale images using a single reference color image. A locality consistent regularization term and an edge-preserving filter are introduced to enhance matching consistency and improve visual quality. Experimental results demonstrate the superiority of the method compared to state-of-the-art techniques in both visual and quantitative evaluations.

F. Ozcelik, U. Alganci, E. Sertel, and G. Unal (2021) [13] proposes a self-supervised learning framework for satellite image pansharpening. Instead of focusing on super-resolution, the approach treats pansharpening as a colorization problem and utilizes a novel PanColorGAN model. By introducing noise injection and adversarial training, the method overcomes spatial-detail loss and blur issues observed in CNN-based pansharpening methods. Experimental results demonstrate the superiority of the proposed approach over previous CNN-based and traditional methods.

G. Ji, Z. Wang, L. Zhou, Y. Xia, S. Zhong, and S. Gong (2021) [14] presents a novel method for colorizing SAR images using a multidomain cycle-consistency GAN (MC-GAN). The approach eliminates the need for paired SAR-optical images and achieves accurate colorization through the use of a mask vector and cycle-consistency loss. Experimental results demonstrate the effectiveness of the proposed method compared to other techniques.

G. Kong, H. Tian, X. Duan, and H. Long (2021) [15] presents a new adversar-

ial edge-aware image colorization method that incorporates semantic information. The proposed method achieves superior results compared to existing approaches by addressing edge color leakage and employing specific loss functions during training. Experimental results validate the effectiveness of the method in image colorization.

Y. Zhao, L.-M. Po, K.-W. Cheung, W.-Y. Yu, and Y. A. U. Rehman (2021) [16] introduces a fully automatic Saliency Map-guided Colorization with Generative Adversarial Network (SCGAN) framework for colorizing grayscale images. The proposed method jointly predicts the colorization and saliency map to minimize semantic confusion and color bleeding. By utilizing pre-trained VGG-16-Gray network and hierarchical discriminators, SCGAN achieves perceptually reasonable colorization with less training data compared to state-of-the-art methods, as demonstrated in evaluations on the ImageNet validation set.

S. Iizuka, E. Simo-Serra, and H. Ishikawa (2016) [17] proposes a deep neural network-based technique for automatic grayscale image colorization that combines global priors and local image features. The network incorporates a fusion layer to merge local information from small image patches with global priors computed using the entire image. The approach is trained end-to-end and can process images of any resolution, achieving significant improvements over the state of the art in terms of realistic and visually appealing colorizations, as validated through a user study and demonstrated across various image types.

K. Nazeri, E. Ng, and M. Ebrahimi (2018) [18] discusses the significance of automatic image colorization and its challenges, particularly in dealing with ill-posed problems and diverse color assignments. The proposed approach focuses on achieving generalization by utilizing a conditional Deep Convolutional Generative Adversarial Network (DCGAN) trained on publicly available datasets. The results of the generative model are compared with traditional deep neural networks, highlighting the effectiveness of the proposed method.

Table 2.1: Review Matrix with Research Papers, Authors, Summary & Year of Publication.

S.N.	**Title**	**Author/s**	**Summary**	**Year**
1	Automated Colorization of a Grayscale Image With Seed Points Propagation [10]	S. Wan, Y. Xia, L. Qi, Y.-H. Yang, and M. Atiquzzaman	uses superpixels, neural network, and optimization to achieve automated colorization with improved accuracy. PSNR:26.18 SSIM:0.83	2020
2	Learning Diverse Image Colorization [11]	A. Deshpande, J. Lu, M.-C. Yeh, M. J. Chong, and D. Forsyth	variational autoencoder (VAE) model with loss terms that address blurry outputs and the uneven distribution of pixel colors, leading to a conditional model	2017

3	Example-Based Image Colorization Using Locality Consistent Sparse Representation [12]	B. Li, F. Zhao, Z. Su, X. Liang, Y.-K. Lai, and P. L. Rosin	example-based image colorization method operating at the superpixel level, utilizing sparse pursuit and a locality consistent sparse representation.	2017
4	Rethinking CNN-Based Pansharpening: Guided Colorization of Panchromatic Images via GANs [13]	F. Ozcelik, U. Alganci, E. Sertel, and G. Unal	PanColorGAN framework using grayscale transformed multispectral images as input, addressing fixed downscale ratio assumptions with noise injection, and employing adversarial training to improve spatial detail.	2021
5	SAR Image Colorization Using Multidomain Cycle-Consistency Generative Adversarial Network [14]	G. Ji, Z. Wang, L. Zhou, Y. Xia, S. Zhong, and S. Gong	colorizing synthetic aperture radar (SAR) images using a multidomain cycle-consistency generative adversarial network MC-GAN, which incorporates a mask vector and multidomain classification loss. PSNR:14.066 SSIM:0.117	2021
6	Adversarial Edge-Aware Image Colorization With Semantic Segmentation [15]	G. Kong, H. Tian, X. Duan, and H. Long	combines adversarial edge-aware image colorization with semantic segmentation, using a deep semantic fusion generator to infer semantic clues. PSNR:31.359 SSIM:0.972	2021
7	SCGAN: Saliency Map-Guided Colorization With Generative Adversarial Network [16]	Y. Zhao, L.-M. Po, K.-W. Cheung, W.-Y. Yu, and Y. A. U. Rehman	framework that combines color prediction with saliency map guidance to minimize semantic confusion and color bleeding. PSNR:23.80 SSIM:0.9473	2021
8	joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification [17]	S. Iizuka, E. Simo-Serra, and H. Ishikawa	deep learning-based approach that combines global priors and local image features. PSNR:29.851 SSIM:0.955	2016

9	Image colorization using generative adversarial networks [18]	K. Nazeri, E. Ng, and M. Ebrahimi	conditional Deep Convolutional Generative Adversarial Network (DCGAN) trained on publicly available datasets to fully generalize the image colorization process. PSNR:29.415 SSIM:0.928	2018

Chapter 3

Requirement Analysis

3.1 SOFTWARE REQUIREMENT

Our image colorization system requires Python,TensorFlow, Django, Keras, Pytorch, Numpy, Jupyterlab, HTML & CSS which are described below;

3.1.1 Python

Python, created by Guido van Rossum in 1991, is a popular and versatile higlevel programming language known for its simplicity and readability. We utilized several libraries and frameworks in this project, including:

- NumPy
- Matplotlib
- OpenCV
- Tensorflow

3.1.2 Google Colab

Google Colaboratory, also known as Google Colab, is a web-based tool developed by Google Research. It enables users to write and run Python code directly in their browser. Colab is particularly useful for tasks like machine learning, data analysis, and educational purposes.

3.1.3 Visual Studio Code

VSCode, also known as Visual Studio Code, is a free and open-source code editor created by Microsoft. It has a lightweight and customizable interface and supports multiple programming languages.

3.1.4 HTML & CSS

HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with

graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications. CSS is the language for describing the presentation of Web pages, including colors, layout, and fonts.

3.2 HARDWARE REQUIREMENT

Some general hardware considerations are:

- Processor (CPU): Intel core i5(or above) or AMD Ryzen 5
- Memory (RAM): At least 8GB
- Graphics Processing Unit (GPU): GTX 1650 or AMD Radeon RX 5500 XT or above

3.3 FUNCTIONAL REQUIREMENT

Functional requirements are the requirements specified by the end user.These requirements define the main functionality of the system or one of its sub-components and describes about the services the system or program should provide to the user. Our system includes the following functional requirements.

3.3.1 Image Detection

The system should be able to take grayscale image as input.

3.3.2 Decode Image

The system should be able to decode image and convert it into matrix form for the future mathematical calculation used for image colorization.

3.3.3 Image Colorization

The system should be able to colorize the given grayscale input image with proper color selections.

3.4 NON-FUNCTIONAL REQUIREMENT

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. A functional requirements define what a system is supposed to do and non-functional requirements define how a system is supposed to be.

3.4.1 Reliability

Reliability is the ability of the system to run without failure. A system will be reliable if all the components of the system including hardware, firmware and software will satisfactorily perform its intended task in the the specified environment for the specified time. Our system will be reliable.

3.4.2 Portability

Portability of a system refers to the ability of a system to be used in different environments without affecting the performance of the system. The system we are developing will be portable over various devices.

3.4.3 Scalability

The system should be scalable to process images in different platform without sticking to certain platform.

3.4.4 Flexibility

The system should be able to handle different image formats.

3.4.5 User Interface

The system should provide a user-friendly interface, allowing users to interact with and control the image colorization process. This could include options for adjusting parameters, selecting specific sources, and visualizing the colorized image.

3.4.6 Low Latency

In scenarios where real-time processing is required, the system should introduce minimal latency to maintain instant result of image colorixation.

3.4.7 Extensibility and Modularity

The system should be designed to be extensible and modular, allowing for easy integration with additional algorithms, techniques, or enhancements. This enables future improvements or incorporation of new research findings in the field of image colorization, ensuring the system stays up-to-date and adaptable to evolving needs and advancements in the computer vision.

3.4.8 Computational Efficiency

The system should be computationally efficient, minimizing the required processing power and memory usage to ensure real-time or near-real-time performance on different hardware platforms.

3.5 FEASIBILITY STUDY

The feasibility of this system can be studied under following points.

3.5.1 Economic Feasibility

There is no need of an actual capital for the completion of the project, as the expenses mainly involve computational resources for model creation and training. This requires a good GPU and memory. The dataset can be obtained through an application approval process which may take time but will not have economic implications as the they are available on the internet. All software chosen for the project is open source and user-friendly. We will utilize our own CPUs and GPUs, as well as Google Colaboratory cloud computing service, for the required computational power. In case of heavy processing, we have access to GPUs provided by our college. Thus, the project is economically fisible.

3.5.2 Technical Feasibility

Technically, this project of using GAN based techniques for image colorization is challenging. Although there may not be ready-made models for this specific task, we can adapt existing GAN architectures for our purposes. We have access to suitable computational resources such as CPUs, GPUs, and cloud computing services like Google Colaboratory. We can use popular frameworks like PyTorch or TensorFlow for implementation, making it easier to integrate the generative models. We can evaluate the performance using specific metrics to assess the quality of the output colorized image. While transformers may take longer to train and have more complexity, we have the potential to achieve satisfactory results in converting grayscale images to colorized image. So, implementing this system seems technically feasible.

3.5.3 Operational Feasibility

The major requirement for our system is coversion of grayscale image to colorized form. To achieve this, we will implement suitable colorization algorithms and train a GAN model to learn the colorization of image. Once the model has been trained and meets our accuracy and loss targets, the project becomes operational. We can then integrate the system into web-based applications or mobile applications, making it easily accessible for individuals to use. The system will be designed to be reliable and maintainable for long-term use, ensuring its operational feasibility. Taking into account these considerations and the potential for future enhancements and updates, we can confidently conclude that this project is operationally feasible.

Chapter 4

System Design and Architecture

We aim to develop a system that accepts grayscale images as input and transforms them into colourful images. The system will utilize various algorithms and techniques to generate images that closely resemble realistic color of image, capturing the details, colors and textures of the subject portrayed in the input grayscale image. This tool will enable to visualize old black and white images into a colourful form.

4.1 Use case Diagram

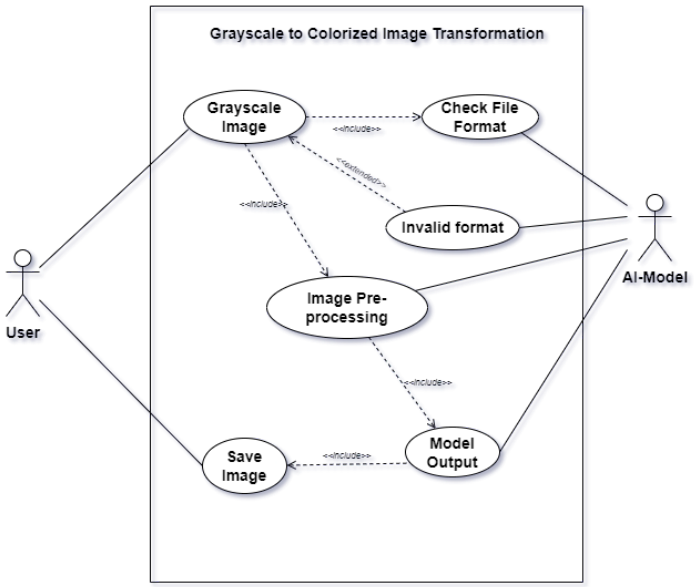


Figure 4.1: Use Case Diagram

The user can input grayscale image in the UI, and that image will undergo preprocessing before being passed to an AI model for output generation. The AI model will generate an output based on the grayscale image. Finally, the user can save the final result.

4.2 System Block Diagram

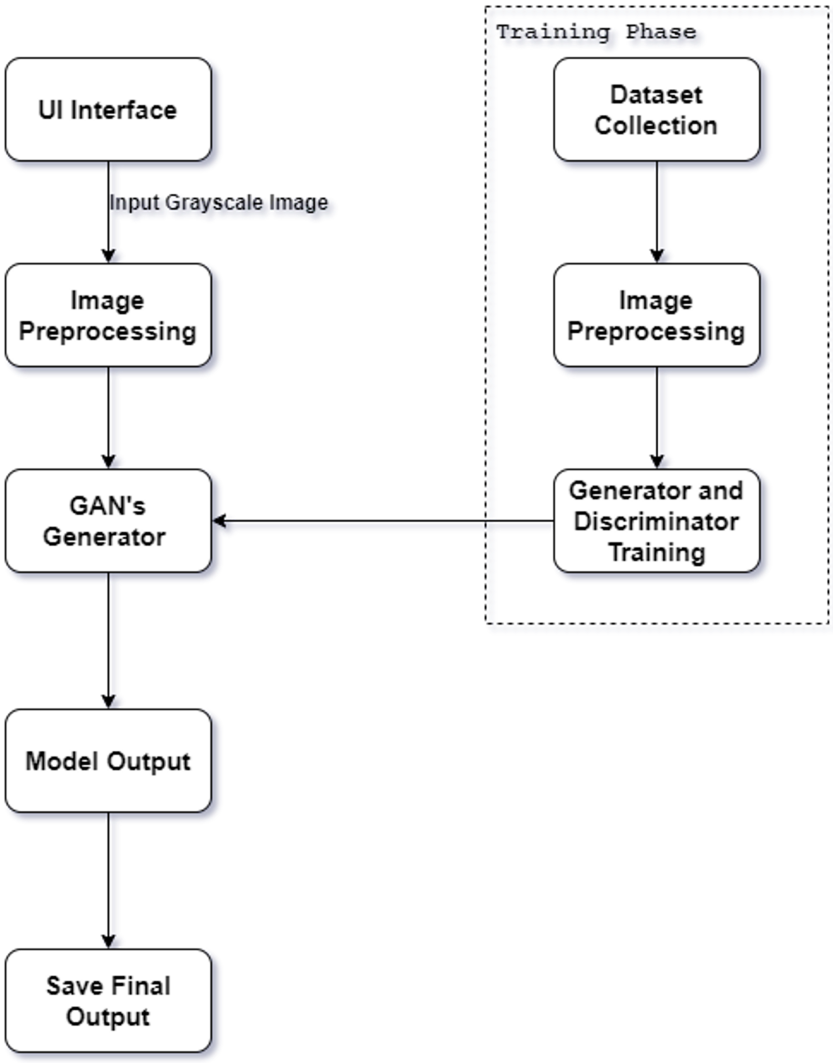


Figure 4.2: System Block Diagram

The user begins by uploading an grayscale image on the UI interface. The image undergoes preprocessing before being fed into an AI model. The model generates an output image based on the input grayscale image. Finally, the user has the option to save the transformed image as the final result for further use or sharing.

Chapter 5

Work Completed

5.1 Data Gathering

Our image colorization project depends on large number of image datasets of old monuments that includes various temples, stupas, landmarks etc. We've successfully collected 11194 images of old monuments which are in colorized form that can be used for training and testing of the proposed model. Collected Images were splitted into training, validation and testing part in the ratio of 7:2:1
We also collected few old images of monuments whose existence are only available in grayscale image. We aim to build a working model that can realistically colorized these old images of various monuments.

Table 5.1: Data for GAN training and evaluation

Collected RGB Image of Monument			**Old Grayscale Image**
Training	**Validation**	**Testing**	
7835	2238	1121	71

5.2 Data Preprocessing

Pre-processing is required in order for the collected images to fit the model building properly and enhance the outcome of the model. We have performed following steps in order for the image preprocessing in the preparation of dataset.

5.2.1 Normalization

All the values of input image are converted into the range of 0 to 1 by mapping value of 0 in pixel to 0 and 255 to 1. This conversion is done by following equation.

$$x_{normalized} = \frac{x}{255.0} \tag{5.1}$$

5.2.2 Resize

For the model building and training process, we decided to take images of size 256x256. It is used for both training input and target. So we converted all collected images into size of 256x256.

5.2.3 Data Augmentation

For the diversity in the collected data, data augmentation is performed in the collected data. Various data augmentation methods used are random crop, rotation, horizontal flipping.

5.2.4 RGB to YUV Conversion

YUV is a color space that separates image luminance (brightness) information (Y) from chrominance (color) information (U and V). In the YUV color scheme, the "Y" stands for luminance or brightness. It represents how bright a color is. If you think of a black-and-white image, the Y component is what determines the grayscale intensity. So, the higher the Y value, the brighter the color.
The "U" and "V" components are responsible for color information. "U" represents the difference between the brightness and the blue component, and "V" represents the difference between the brightness and the red component. In other words, they help define the color tone. If "U" is high, it means there's more blue, and if "V" is high, there's more red.
So, if we have a pixel described in YUV, it could be something like: "This pixel is quite bright (high Y), has a bit more blue than green (higher U), and has a touch more red than blue (higher V)."
In our project, the Y-channel is used as the input for the pix2pix model, and the goal is to predict U and V channel as realistically as possible. In this case only 2 channels are required to be generated from the GAN generator whereas in the case of RGB system, all 3 channels are required to be correctly generated which has much complexity than YUV system.
The conversion formulas from RGB to YUV and vice versa involve a set of mathematical operations. Here are the standard formulas:

5.2.4.1 RGB to YUV Conversion

$$Y = 0.299R + 0.587G + 0.144B \tag{5.2}$$

$$U = -0.14713R - 0.288862G + 0.436B \tag{5.3}$$

$$V = 0.615R - 0.51498G - 0.10001B \tag{5.4}$$

5.2.4.2 YUV to RGB Converion

$$R = Y + 1.13983V \tag{5.5}$$

$$G = Y - 0.39465U - 0.5806V \tag{5.6}$$

$$B = Y + 2.03211U \tag{5.7}$$

5.3 Prepare Model

To generate colorized image from the input grayscale image, we used image to image translation GAN architecture i.e. pix2pix cGAN is mainly used to generate some images from equivalent image as input which is translated through encoder and decoder during the process. This GAN contained 2 main components which are Generator and Discriminator.

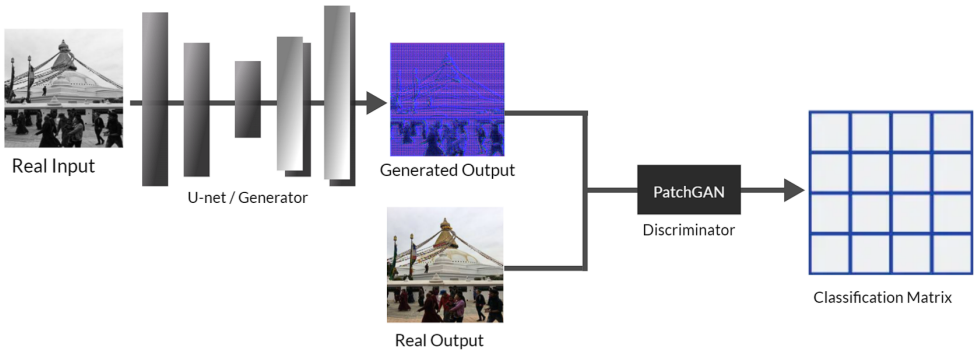


Figure 5.1: Pix2pix Model

5.3.1 Generator

The generator in the Pix2Pix cGAN model consists of an encoder-decoder architecture. The encoder takes the grayscale input image and progressively reduces its spatial dimensions while increasing its feature depth. This encoded representation is then passed to the decoder, which upscales and transforms it to generate the final colored image. Skip connections are typically used to connect corresponding encoder and decoder layers, helping to preserve fine details.
The generator's goal is to create colorized images that are convincing and indistinguishable from real colored images. To achieve this, it's trained alongside a discriminator, which is a separate neural network. The discriminator's job is to differentiate between real colored images and those produced by the generator. The generator aims to improve its performance by trying to generate images that can fool the discriminator into believing they are real.

5.3.2 Discriminator

The PatchGAN discriminator is a key component of the Pix2Pix model used for image-to-image translation tasks like image colorization. Unlike traditional discriminators that classify the entire input image as real or fake, the PatchGAN discriminator focuses on making judgments at the patch level. This allows it to provide fine-grained feedback on the realism of different parts of the generated image. PatchGAN discriminator works by sliding a small window (patch) over both the real target images and the images generated by the generator. For each patch, the discriminator produces a probability score indicating whether the patch is real or fake. These probability scores are used to compute a loss that guides the training process.
The advantage of using a PatchGAN discriminator lies in its ability to provide spatially detailed feedback. Instead of generating a single probability for the entire image, it generates a grid of probabilities, effectively allowing the model to capture local details and textures. This is particularly useful for tasks like image colorization, where the quality of colorization can vary significantly across different regions of an image.
The PatchGAN discriminator's outputs can be thought of as a heatmap of probabilities, indicating the likelihood that each patch in the input image is real or

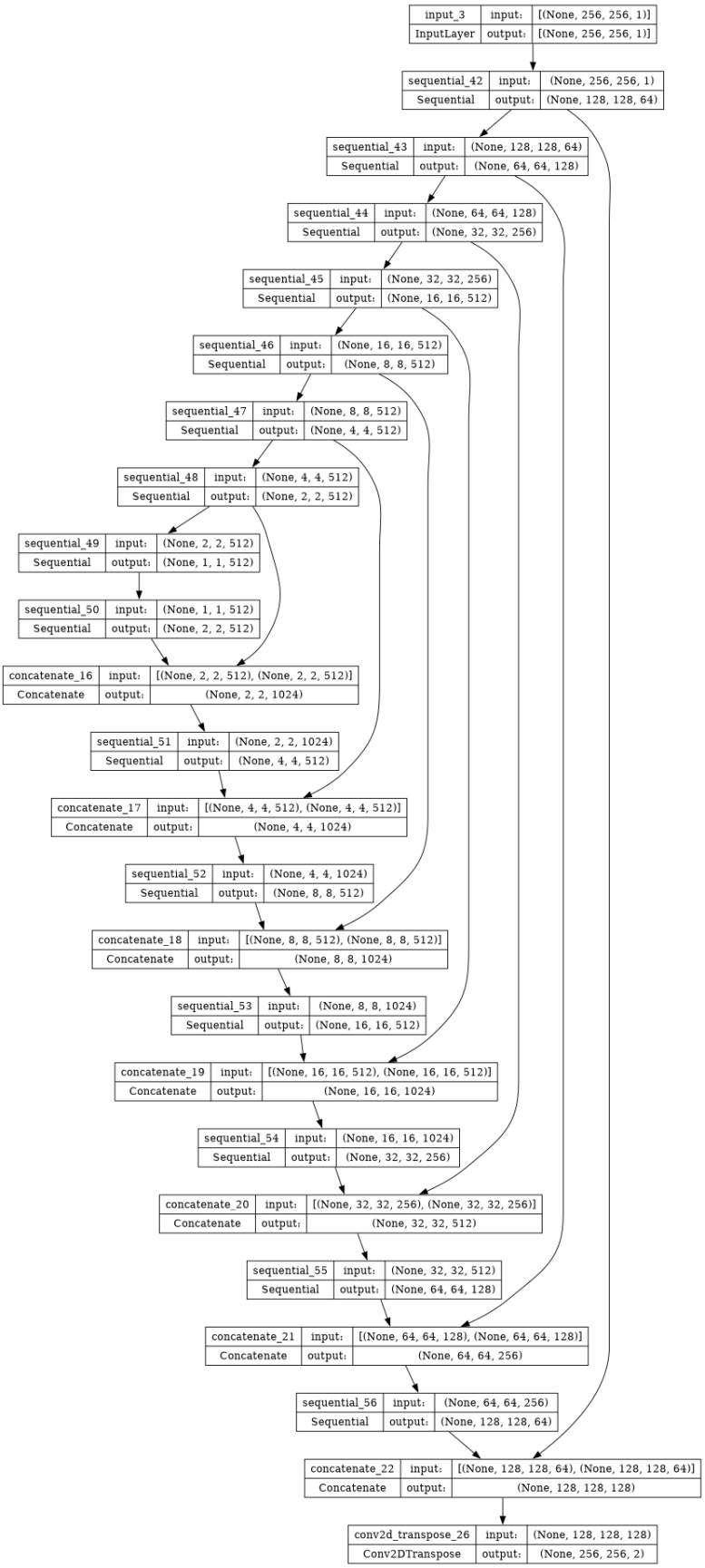


Figure 5.2: U-net Generator in pix2pix

fake.

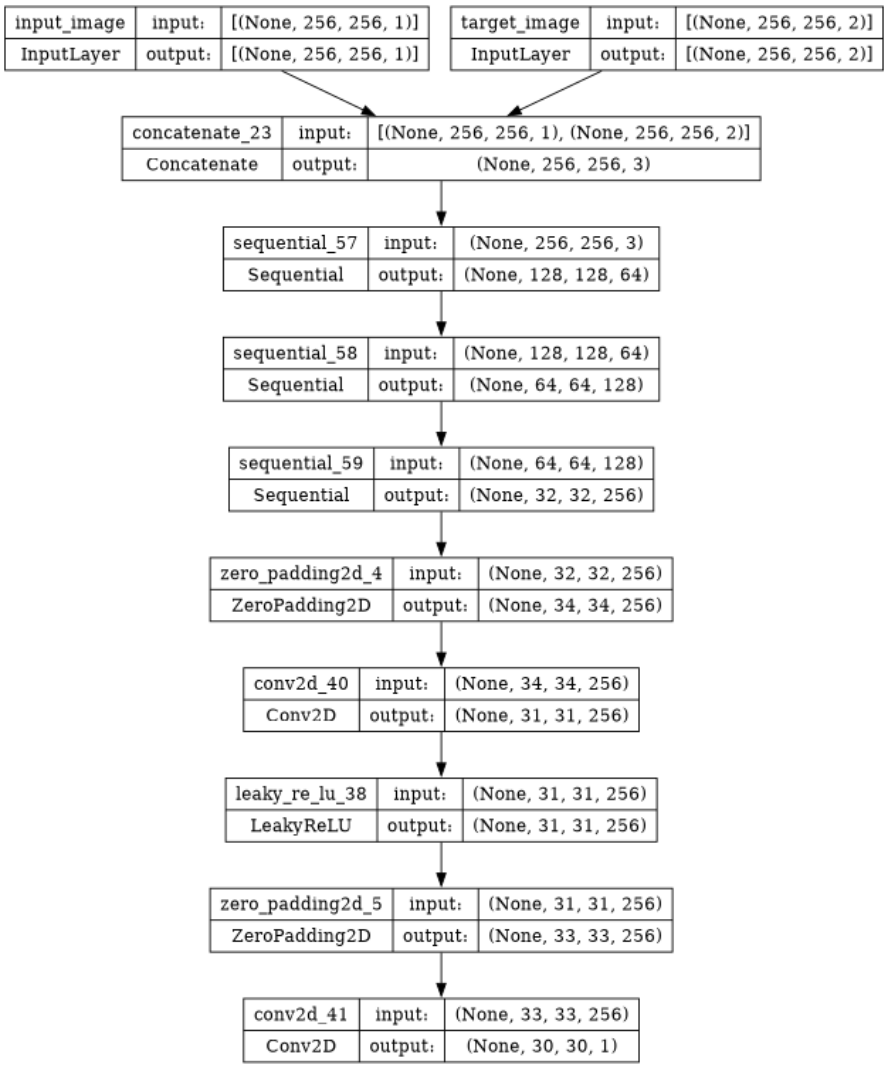


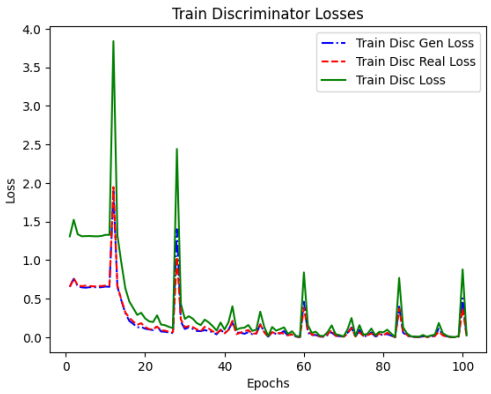
Figure 5.3: PatchGAN Discriminator in pix2pix

5.4 Model Evaluation

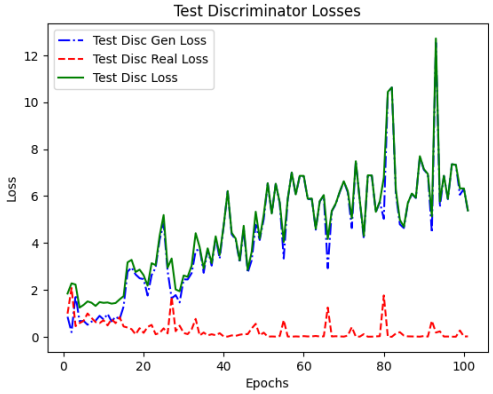
After the model is defined and trained, the model is evaluated based on different generator and discriminator losses.

5.4.1 Discriminator loss

Discriminator loss is typically computed using a binary cross-entropy loss function. This loss function measures the difference between the predicted probabilities by the discriminator for real and generated images and the actual labels. This also had 2 components which are:



(a) Discriminator Loss for Training Data



(b) Discriminator Loss for Validation Data

5.4.1.1 Discriminator Loss for Real Samples

The discriminator's task is to correctly classify real samples (images from the true dataset) as real. The loss for real samples is calculated based on how well the discriminator's predictions match the target labels (which are all ones, indicating real samples).Mathematically, the loss for real samples is given by:

$$L_{real} = -\frac{1}{N}\sum_{i=1}^{N} logD(x_i) \tag{5.8}$$

where, N is the number of samples in current batch.
x_i representss the i-th real sample
$D(x_i)$ is the discriminator's predicted probability that x_i is a real sample.

5.4.1.2 Discriminator's Loss for Generated Samples

The discriminator's second task is to correctly classify generated samples (images produced by the generator) as fake. The loss for generated samples is calculated based on how well the discriminator's predictions match the target labels. Mathematically, the loss for generated samples is given by:

$$L_{generated} = -\frac{1}{N}\sum_{i=1}^{N} log(1 - D(G(z_i))) \tag{5.9}$$

where, N is the number of samples in current batch.
z_i is the number of samples in current batch.
$G(z_i)$ is the i-th generated sample produced by the generator using z_i
$D(G(z_i))$ is the discriminator's predicted probability that $G(z_i)$ is a real sample (which the generator aims to make as low as possible).

$$L_{discriminator} = L_{real} + L_{generated} \tag{5.10}$$

5.4.2 Generator Loss

the generator's primary objective is to produce images that are indistinguishable from real images to the discriminator. The generator loss is a key part of achieving this goal. The loss guides the generator's learning process by quantifying how well it is able to fool the discriminator. The loss typically involves two parts:

5.4.2.1 Adversarial Loss

The adversarial loss, often referred to as the "generator adversarial loss," is based on how well the generator can deceive the discriminator. It is calculated using the predicted probabilities of the discriminator for the generated images. The generator aims to minimize this loss by producing images that the discriminator cannot confidently classify as fake. Mathematically, the adversarial loss is given by:

$$L_{adv} = -\frac{1}{N}\sum_{i=1}^{N} log(D(G(z_i))) \tag{5.11}$$

where, N is the number of samples in current batch.
z_i is the number of samples in current batch.
$G(z_i)$ is the i-th generated sample produced by the generator using z_i
$D(G(z_i))$ is the discriminator's predicted probability that $G(z_i)$ is a real sample (which the generator aims to make as low as possible).

5.4.2.2 L1 loss

The L1 loss, also known as the mean absolute error (MAE) loss, is a commonly used loss function in machine learning and deep learning tasks, including image-to-image translation models like Pix2Pix. It measures the average absolute difference between the predicted values and the ground truth values. In the context of Pix2Pix and other image translation tasks, L1 loss is often used to ensure that the generated images closely match the target images. L1 loss encourages the generator to produce images that closely match the ground truth images in terms of pixel values. This helps ensure that the generated colorized images exhibit accurate colors and details, promoting realistic and visually pleasing results. Mathematically, the L1 loss between two sets of data points y_i and $\hat{y}_i$ is calculated as:

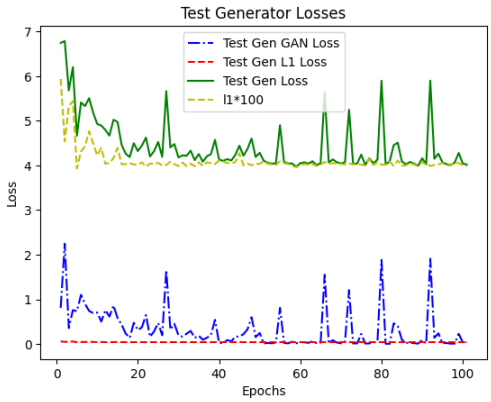
$$L_{L1} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{5.12}$$

where, N is the number of data points (usually pixels) in the images.
y_i represents the ground truth value at data point i.
$\hat{y}_i$ represents the predicted value at data point i (in the context of Pix2Pix, this would be the value generated by the model).

$$L_{generator} = L_{adv} + \lambda . L_{L1} \tag{5.13}$$

Here, λ is a hyperparameter that balances the importance of the adversarial loss and the additional losses. It determines how much emphasis is placed on fooling the discriminator versus other objectives. For our model we took $\lambda = 100$

(a) Generator Loss for Training Data



(b) Generator Loss for Validation Data

5.4.3 SSIM Evaluation

SSIM index produces a value between -1 and 1, where 1 indicates that the two images are identical, and a higher value generally implies better similarity. We calculated SSIM score for one image after each epoch and the result was as in 5.7a. We also compared SSIM score for YUV colour scheme and RGB colour scheme. And the result for YUV colour scheme were much superior than the RGB ones5.6.

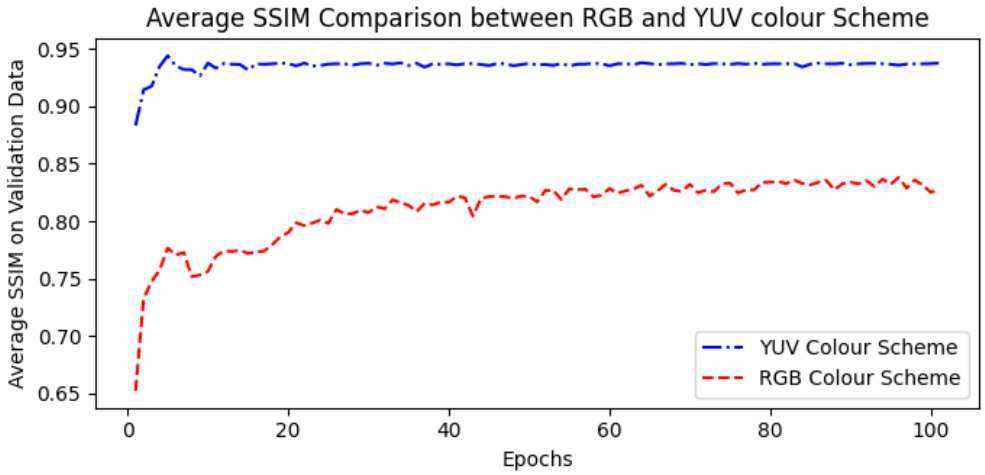
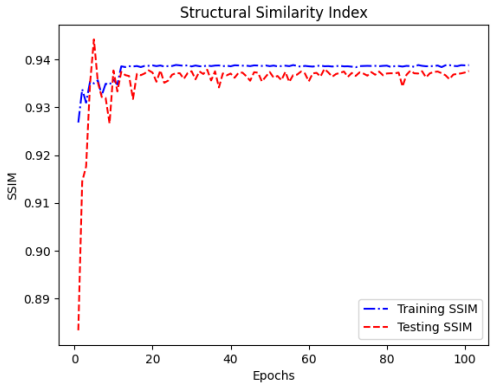


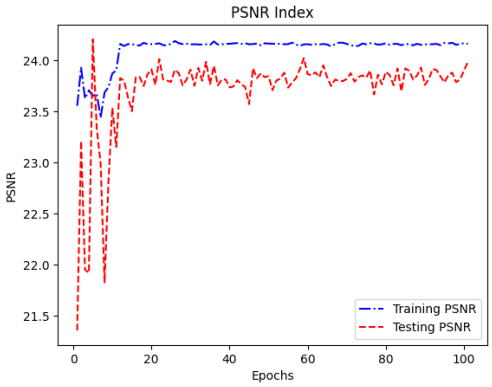
Figure 5.6: SSIM comparison between RGB and YUV Colour Scheme

5.4.4 PSNR Evaluation

PSNR, or Peak Signal-to-Noise Ratio, is a metric commonly used to evaluate the quality of reconstructed or compressed images or videos. It measures the fidelity of the reconstructed signal by comparing it to the original, uncompressed signal. The higher the PSNR value, the better the quality of the reconstructed signal. Average PSNR of our model calculated per epoch was as in 5.7b



(a) Average SSIM Evaluation



(b) Average PSNR Evaluation

5.5 Run Test Samples

We did validation test on few test cases. Few results are shown below.

Figure 5.8: validation test image 1

Figure 5.9: validation test image 2

Figure 5.10: validation test image 3

We also run trained model to generated colorized image for some old images of monuments. And results were as follows:

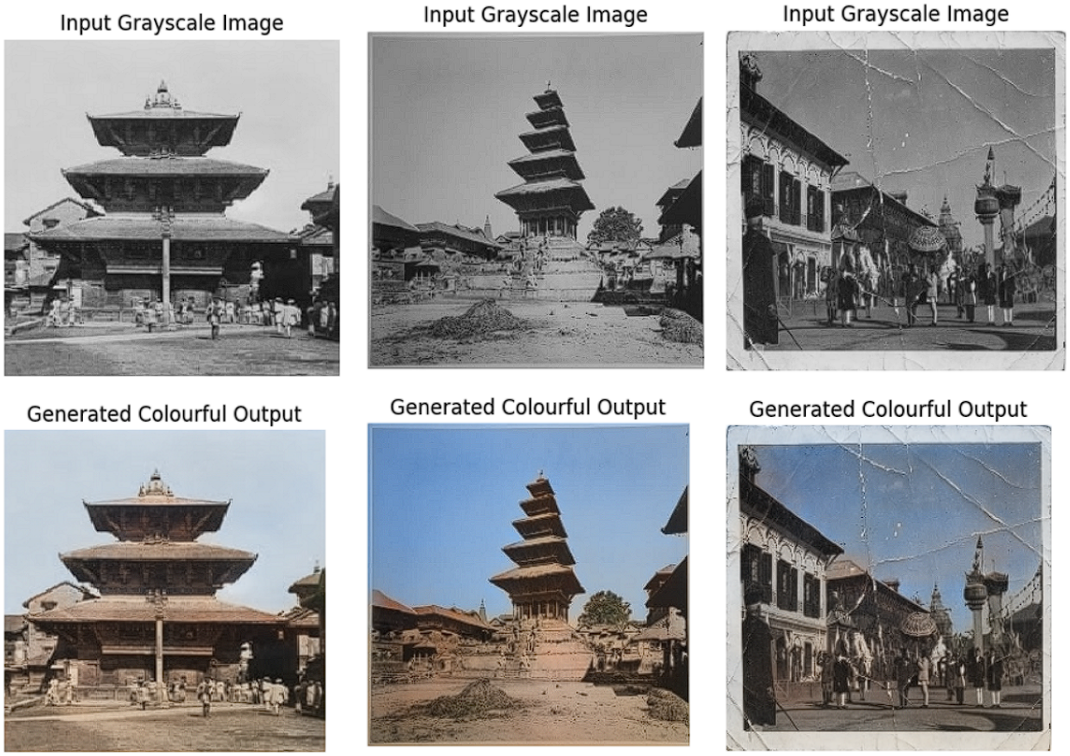


Figure 5.11: Test run on Old Grayscale Images

Chapter 6

Work in Progress

6.1 Model Refinement

The model we built is not able to produce enough accurate result. We will have to study more on how to increase the accuracy of the model and refine it for the better model building.

6.2 Model Evaluation on Benchmark Dataset

We haven't yet tested this model on benchmark datasets for image colorization. It will help us compare our models to present state of art models and help in improving the model.

6.3 Model Deployment

The deployment of the model is going to be on web where user can upload grayscale image and see colorized output image which they can download as well.

Bibliography

[1] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *IEEE transactions on image processing*, vol. 15, no. 5, pp. 1120–1129, 2006.

[2] P. Vitoria, L. Raad, and C. Ballester, "Chromagan: Adversarial picture colorization with semantic class distribution," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2445–2454.

[3] Y. Wu, X. Wang, Y. Li, H. Zhang, X. Zhao, and Y. Shan, "Towards vivid and diverse image colorization with generative color prior," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 14 377–14 386.

[4] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002, pp. 277–280.

[5] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Computer graphics and applications*, vol. 21, no. 5, pp. 34–41, 2001.

[6] G. Zong, Y. Chen, G. Cao, and J. Dong, "Fast image colorization based on local and global consistency," in *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*, vol. 1. IEEE, 2015, pp. 366–369.

[7] C. Gauge and S. Sasi, "Automated colorization of grayscale images using texture descriptors and a modified fuzzy c-means clustering," 2012.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets (advances in neural information processing systems)(pp. 2672–2680)," *Red Hook, NY Curran*, 2014.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[10] S. Wan, Y. Xia, L. Qi, Y.-H. Yang, and M. Atiquzzaman, "Automated colorization of a grayscale image with seed points propagation," *IEEE Transactions on Multimedia*, vol. 22, no. 7, pp. 1756–1768, 2020.

[11] A. Deshpande, J. Lu, M.-C. Yeh, M. J. Chong, and D. Forsyth, "Learning diverse image colorization," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2877–2885.

[12] B. Li, F. Zhao, Z. Su, X. Liang, Y.-K. Lai, and P. L. Rosin, "Example-based image colorization using locality consistent sparse representation," *IEEE Transactions on Image Processing*, vol. 26, no. 11, pp. 5188–5202, 2017.

[13] F. Ozcelik, U. Alganci, E. Sertel, and G. Unal, "Rethinking cnn-based pansharpening: Guided colorization of panchromatic images via gans," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 4, pp. 3486–3501, 2021.

[14] G. Ji, Z. Wang, L. Zhou, Y. Xia, S. Zhong, and S. Gong, "Sar image colorization using multidomain cycle-consistency generative adversarial network," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 2, pp. 296–300, 2021.

[15] G. Kong, H. Tian, X. Duan, and H. Long, "Adversarial edge-aware image colorization with semantic segmentation," *IEEE Access*, vol. 9, pp. 28 194–28 203, 2021.

[16] Y. Zhao, L.-M. Po, K.-W. Cheung, W.-Y. Yu, and Y. A. U. Rehman, "Scgan: Saliency map-guided colorization with generative adversarial network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 8, pp. 3062–3077, 2021.

[17] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color! joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (ToG)*, vol. 35, no. 4, pp. 1–11, 2016.

[18] K. Nazeri, E. Ng, and M. Ebrahimi, "Image colorization using generative adversarial networks," in *Articulated Motion and Deformable Objects: 10th International Conference, AMDO 2018, Palma de Mallorca, Spain, July 12-13, 2018, Proceedings 10*. Springer, 2018, pp. 85–94.