IMAGE MANIPULATION USING GENERATIVE ADVERSARIAL NETWORKS

A MINI PROJECT REPORT

Submitted by

VISHAL J C (2127200501164)
VISHAL M (2127200501165)
VIVEK MUNNAA D (2127200501169)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

SRI VENKATESWARA COLLEGE OF ENGINEERING

(An Autonomous Institution; Affiliated to Anna University, Chennai-600025)

ANNA UNIVERSITY :: CHENNAI 600 025

DECEMBER 2023

SRI VENKATESWARA COLLEGE OF ENGINEERING
(An Autonomous Institution; Affiliated to Anna University, Chennai-600025)

ANNA UNIVERSITY :: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"IMAGE MANIPULATION USING GENERATIVE ADVERSARIAL NETWORKS"** is the bonafide work of **"VISHAL J C (2127200501164), VISHAL M (2127200501165)** and **VIVEK MUNNAA D (2127200501169)"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.R.ANITHA	**Ms.P.UMA**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
	ASSISTANT PROFESSOR
COMPUTER SCIENCE & ENGG	**COMPUTER SCIENCE & ENGG**

Submitted for the project viva-voce examination held on ……..

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Image colorization, the process of adding color to grayscale images, has been a subject of fascination and research for several years. This thesis explores the application of Conditional Generative Adversarial Networks (CGANs) to the field of image colorization, aiming to enhance and innovate the process. The research investigates the potential of CGANs to generate colorized images, given grayscale inputs, leveraging the power of conditional models. We propose a novel approach that employs a U-Net architecture for the generator and a PatchGAN discriminator for adversarial learning. The model is designed to go beyond mere colorization, enabling the generation of realistic and high-quality colorized images that preserve fine details and textures. The outcomes of our research highlight the potential of our method in various domains, including computer vision, art restoration, and image enhancement and also delves into the definition of GAN loss functions and their role in the training process.

ACKNOWLEDGEMENT

We thank our Principal **Dr. S. Ganesh Vaidyanathan,** Sri Venkateswara College of Engineering for being the source of inspiration throughout our study in this college.

We express our sincere thanks to **Dr. R. Anitha,** Head of the Department, Computer Science and Engineering for her encouragement accorded to carry this project.

With profound respect, we express our deep sense of gratitude and sincere thanks to our guide **Ms. P. Uma**, **Assistant Professor**, for her valuable guidance and suggestions throughout this project.

We are also thankful to our Mini Project coordinators **Dr. R. Jayabhaduri, Professor, Dr. N. M. Balamurugan, Professor** and **Dr. N. Revathi, Associate Professor** for their continual support and assistance.

We thank our family and friends for their support and encouragement throughout the course of our graduate studies.

VISHAL J C
VISHAL M
VIVEK MUNNAA D

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

GAN	Generative Adversarial Networks
DCGAN	Double Channel GAN
CGAN	Conditional GAN
HiGAN	Hierarchical GAN
CNN	Convolutional Neural Networks
FID	Fréchet inception distance
WGAN	Wasserstein GAN
ReLU	Rectified Linear Unit
LSGAN	Least Squares GAN
DCNN	Deep Convolutional Neural Network
CVPR	Computer Vision and Pattern Recognition
PCA	Principle Component Analysis
DLMIA	Deep Learning Medical Image Analysis
KNN	K-Nearest Neighbour
LD	Linear Discriminant
SCGAN	Saliency map guided Colorization with GAN

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

In recent decades, the realm of image generation has witnessed remarkable advancements, owing its progress to the continuous evolution of Generative Adversarial Networks (GANs). These strides have significantly enhanced the quality and diversity of generated images, moving beyond the early iterations of GANs like DCGAN to the forefront of innovation, represented by the revolutionary StyleGAN2. Through the process of adversarial training, the generator and discriminator components have been meticulously fine-tuned, shaping the generator into a pre-trained feed forward network, endowed with the extraordinary capability to create images when provided with a randomly sampled vector drawn from a specific distribution.

Nonetheless, the image generation process grapples with inherent limitations. It does not readily accommodate user-driven customizations, such as the transformation of images into different visual styles, like converting a photograph into a painting, or adding personal touches to thegenerated content.

One of the most exciting applications of deep learning is the colorization of black and white images. This task once required extensive human intervention and hardcoding a few years ago. However, today, the entire process can be seamlessly executed end-to-end, thanks to the power of AI and deep learning. While one might assume that training a model for this task necessitates a vast amount of data and extended training times, recent breakthroughs in deep

learning have changed this perception. Furthermore, the complexities surrounding the creation of a convincingly realistic image from the intricate layer-wise representations within the generator continue to pose intriguing challenges. This knowledge gap underscores the imperative need to unravel the learned representations embedded within the depths of deep generative models. This understanding is not mere intellectual curiosity; it is an essential stepping stone to unlock the boundless potential inherent in captivating applications, particularly in the domains of interactive image editing and style transfer. This introductory chapter lays the foundation for our exploration of Generative Adversarial Networks (GANs) and their multifaceted approaches. The comprehension of these intricate models transcends mere academic interest; it is an essential prerequisite to fully harness their capabilities across a broad spectrum of applications.

Moreover, this chapter serves as a prelude to specific focus on image colorization as an intriguing and compelling application of GANs. As we navigate through this thesis, we will delve into the intricacies of howGANs can be harnessed to infuse vibrant color and lifelike realism into grayscale images, opening doors to exciting possibilities in the realm of digital image processing.

1.2 GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks, or GANs, represent a generative model based on deep learning principles. In a broader sense, GANs establish a model framework for training generative models, with a prevalent utilization of deep learning models within this framework.

The GAN model architecture comprises two integral components: a generator model responsible for producing new, plausible instances, and a discriminator model designed to categorize generated instances as either

authentic (belonging to the problem domain) or counterfeit (generated by the generator model).

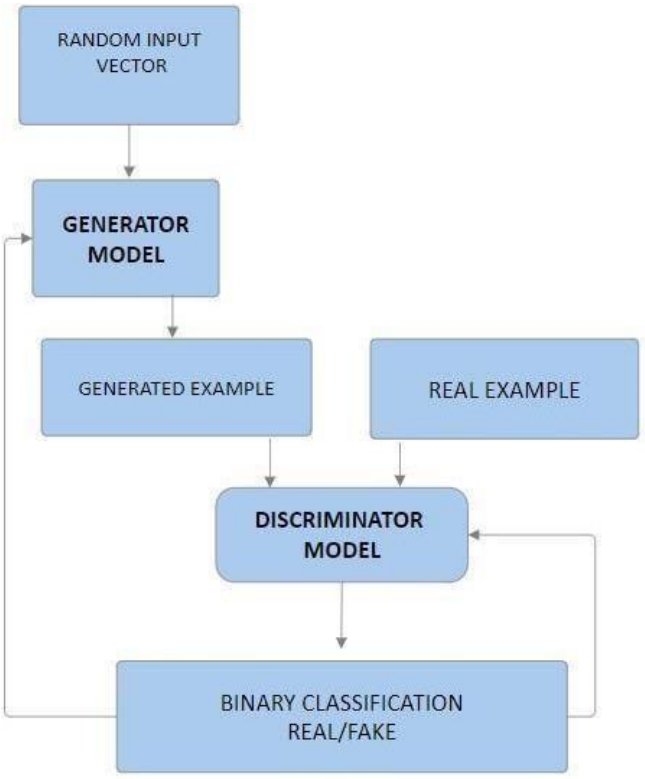

Figure 1.1 – Example of GAN model architecture

Generator component serves as a model that generates credible instances within the problem domain while the discriminator functions as a model for classifying instances as either authentic (representative of the domain) or synthetic (produced by the generator). Generative adversarial networks are grounded in a game-theoretic scenario wherein the generatornetwork engages in a competition with its adversary, the discriminator network. The generator network is tasked with directly creating samples, while the discriminator network strives to differentiate between samples drawn from the training data and those generated by the generator.

1.2.1 THE GENERATOR MODEL

The generator model accepts a fixed-length random vector as input and proceeds to generate a domain-specific sample. This vector is randomly drawn from a Gaussian distribution and acts as the seed for the generative process. Following training, points in this multi-dimensional vector space correspond to points within the problem domain, thereby forming a condensed representation of the data distribution. This vector space is commonly referred to as the latent space, housing latent variablesthat are vital for the domain but remain unobservable.

1.2.2 THE DISCRIMINATOR MODEL

The discriminator model is responsible for receiving an input example, whether real or generated, and providing a binary classification label: real or fake (generated). The real examples originate from the training dataset, while the enerated instances are produced by the generator model. The discriminator operates as a conventional classification model, with its primary purpose completed post-training. Theprimary focus subsequently shifts to the generator. In some cases, the generator can find repurposing applications due to its proficiency in feature extraction from domain-specific examples. Thismay involve the utilization of some or all feature extraction layers in transfer learning endeavors employing similar or identical input data.

1.2.3 CONDITIONAL GANs

Conditional GANs, an important extension of the GAN framework,enable the generation of outputs based on specific conditions. The generative model is trained to produce new instances within the input domain. It operates by taking a random vector from the latent space and generating outputs that are influenced by additional input.

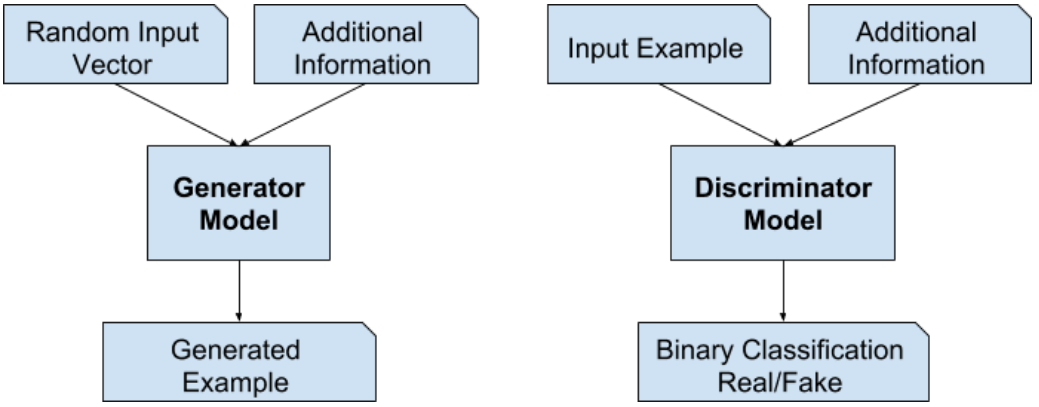

Figure 1.2 Example of Conditional GANs architecture

This supplementary input may consist of class labels, such as "male"or "female" for generating images of people, or numerical values, particularly in tasks like creating images of handwritten digits. The discriminator in conditional GANs is also conditioned, meaningit receives both an input image (real or fake) and the additional input information. For instance, in cases with classification labels as conditionalinput, the discriminator anticipates that the input belongs to a certain class. This dynamic encourages the generator to produce examples aligned withthe specified class, ultimately deceiving the discriminator.

Conditional GANs can be effectively employed to generate instances belonging to a specific category within a domain. Moreover, GAN models can take conditioning a step further by using an existing example from the domain, such as an image, as a reference. This feature enables various applications, including text-to-image and image-to-image translations. This functionality facilitates impressive applications of GANs, like style transfer, photo colorization, or the transformation of images from one season or time of day to another. In scenarios involving conditional GANs for image-to-image translation, such as converting day images to night, the discriminator is provided with both real and generated nighttime images and, conditioned on, real daytime photos as inputs. Meanwhile, the generator utilizes a random vector from the

latent space and, conditioned on, real daytime images for its operation.

1.3 NEED FOR GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks (GANs) have emerged as a pivotal innovation in the realm of deep learning, particularly within domains like computer vision. They offer a versatile approach, with one of the core techniques being data augmentation. Data augmentation is a practice that not only enhances model performance but also introduces a regularization effect that mitigates generalization errors. This technique revolves aroundthe creation of artificial yet credible instances, originating from the problem domain on which the model is trained. In the case of image data, data augmentation often entails basic operations like cropping, flipping, zooming, and other fundamental transformations applied to the existing images in the training dataset. However, GANs bring forth a more advanced and domain-specific approach to data augmentation, going beyond the traditional techniques.

Complex domains or those with limited data availability greatly benefit from the incorporation of generative modeling, offering an extended scope for model training. GANs prove particularly valuable in domains with data scarcity, providing a compelling solution to such challenges. The interest and significance of GANs in research are underscored by their ability to model high-dimensional data effectively, handle missing data, and provide outputs that encompass multiple modes or plausible solutions to a given problem. Conditional GANs represent one of the most compelling applications, specifically designed for tasks that involve the generation of novel examples. These applications include image super-resolution, whichenhances image quality, artistic creation, allowing for the generation of novel and artistic images, and image-to-image translation, facilitatingdiverse image transformations across domains. The captivating aspect of GANs lies in their remarkable success, with the ability to generate imagesof

such realism that they often defy human observers' attempts to distinguish them from real-life photographs.

1.4 MACHINE LANGUAGE BASED APPROACHES

In the ever-evolving landscape of deep learning and generative modeling, the quest for versatile and effective learning approaches has led to significant developments in the field of image manipulation. This chapter delves into the rich tapestry of learning paradigms applied to Generative Adversarial Networks (GANs). We explore two fundamental learning types: supervised, unsupervised each offering unique perspectives and capabilities for harnessing GANs to manipulate and generate images. By understanding the nuances of these learning approaches, we can uncover new dimensions of image manipulation and forge innovative paths in the realm of interactive and customizable image synthesis.

1.4.1 SUPERVISED LEARNING

The supervised strategy harnesses labeled data and well-trained classifiers to probe the latent representations within the generator. An early instance of such an approach is GAN Dissection, which draws inspiration from theprior Network Dissection methodology. GAN Dissection's primary objective is the visualization and comprehension of individual convolutional filters, commonly referred to as "units," embedded within the pre-trained generator. This technique utilizes semantic segmentation networks to segment the generated images and subsequently measures thealignment between the spatial distribution of unit activation maps and the semantic mask applied to the generated image. GAN Dissection has the capability to pinpoint a cluster of interpretable units closely associated with object concepts. These interpretable units can thenfunction as semantic switches, enabling the addition or removal of objects,such as trees or lamps, by adjusting the activation of the corresponding units. In addition to

inspecting the individual generator units, exploratory endeavors extend to the latent space. This is where we select the latent vector for input into the generator, unraveling the latent space's concealed,interpretable dimensions. In this context, the pre-trained generator is denoted and a random vector sampled from the latent space culminate in the output image. As distinct vectors are utilized, the produced images assume distinct forms. Thus, the latent space becomes a repository for diverse image attributes. If we can pinpoint latent subspaces aligned with human-comprehensible concepts, it becomes feasible to steer the image creation process by adjusting the latent code along these pertinent subspaces.

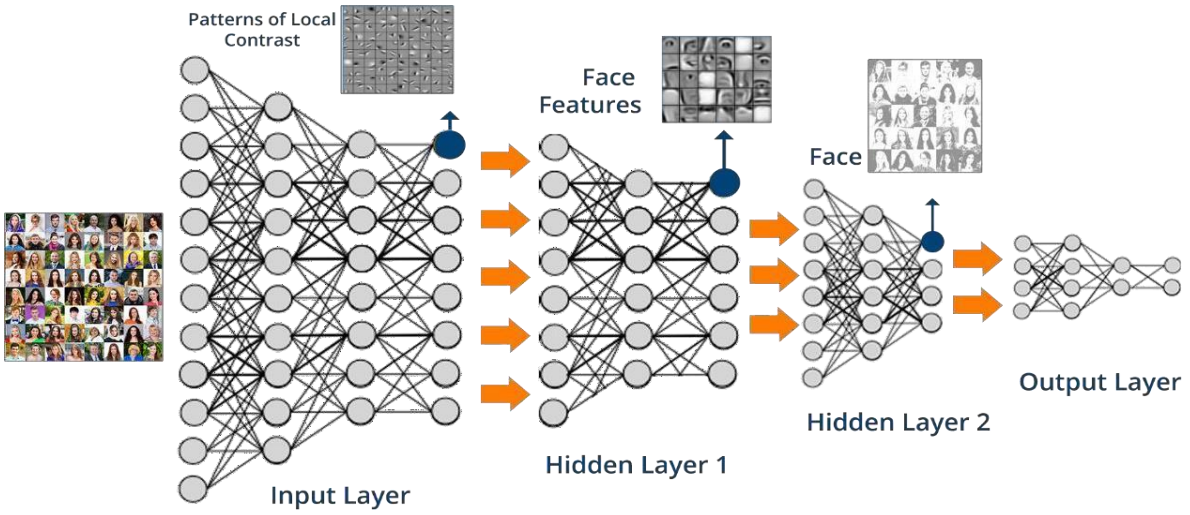

Figure 1.3 Using Contrast to identify Colors

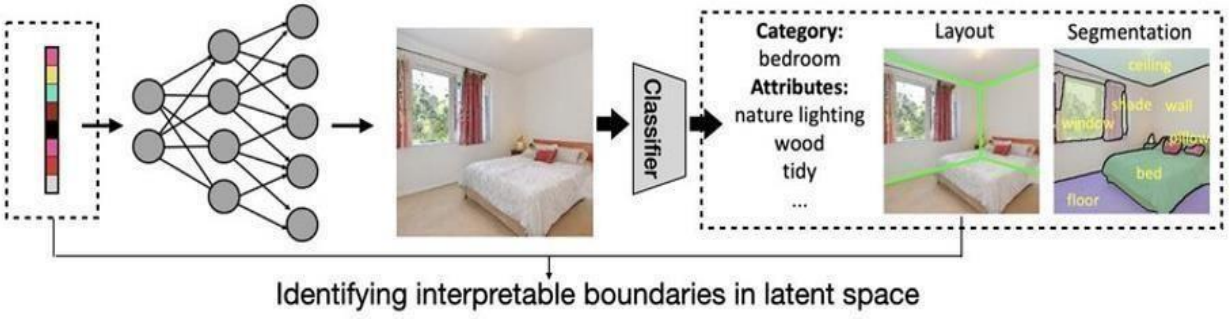

Figure 1.4 Exploring attribute boundaries

To establish a connection between the latent and semantic spaces, we can

utilize off-the-shelf classifiers to extract the characteristics from the generated images, enabling the assessment of causal relationships betweenthe emerging attributes within the generated images and their corresponding vectors in the latent space. The HiGAN method, as proposedin and illustrated in Figure 1.4, aligns with such a supervised approach. This method entails a multi-step process, commencing with the sampling of thousands of latent vectors for image generation. Subsequently, predictions regarding various attribute levels within the generated images are made by applying pre-existing classifiers. These attributes are then used to train linear boundaries within the latent space, employing predicted labels and latent vectors. Finally, a counterfactual verification step is employed to select a reliable boundary. In this process, a linear manipulation model is employed to effect variations in the latent code, facilitating nuanced adjustments in the generated images.

1.4.2 UNSUPERVISED LEARNING

As generative models gain popularity, they find application across diverse datasets, including generating images of cats and anime characters. However, for such disparate data types, the availability of ready-made classifiers can be limited. An alternative path is offered by the unsupervised approach, which seeks to discern the controllable aspects of the generator without relying 11 on predefined labels. This objective revolves around identifying dimensions within the model that, upon projection, induce substantial variations. These identified dimensions correspond to the eigenvectors of the underlying matrix. Different layers of these eigenvectors exercise control over various attributes of the generated image. Manipulating the latent code along these directions facilitates interactive image editing. By uncovering these pivotal dimensions through the decomposition of model weights, GANs can enable interactive content editing.

CHAPTER 2

LITERATURE REVIEW

2.1 INTERPRETING GENERATIVE ADVERSARIAL NETWORKS FOR INTERACTIVE IMAGE GENERATION

Bolei Zhou

Bolei Zhou presented a summary of recent works on interpreting deep generative models. He outlined three interpretation approaches: supervised, unsupervised, and zero-shot, offering practical insights and methods for controlling and customizing image attributes. By bridging the gap between latent spaces and interpretable concepts, this work contributes to the advancement of interactive image editing in the field of deep learning.

2.2 DOUBLE - CHANNEL GUIDED GENERATIVE ADVERSARIAL NETWORK FOR IMAGE COLORIZATION

Kangning Du, Changtong Liu, Lin Cao, Yanan, Fan Zhang and Tao Wang

The paper introduces a DCGGAN network to address abnormal colorization issues in deep learning-based image colorization methods. By incorporating a reference component matching module and a double-channel guided colorization module. The proposed framework is implemented by using the PyTorch deep learning library. The batch size is set to 8 in the stage of

training. The model is trained using the optimizer with $\beta 1 = 0.5$ and $\beta 2 = 0.999$. The initial learning rate is set to 0.0001.

The trade-off hyperparameter λ an ρ between the generative adversarial loss and the component constraint loss are set to 0.01. Results provide an effective solution for achieving accurate and realistic colorization while minimizing abnormal colour artifacts. Through experiments on the different datasets, it illustrates that our approach generates the image with reasonable colour with increased accuracy.

2.3 IMAGE COLORIZATION WITH GENERATIVE ADVERSARIAL NETWORKS

K.Nazeri and E.Ng

Nazeri and Ng explored grayscale image colorization using generative adversarial networks (GANs), comparing results with existing convolutional neural networks (CNNs). They employed a novel cost function for the generator to address training challenges. Preliminary results on CIFAR-10 demonstrate GAN's ability to produce vibrant and visually appealing colorizations. Later, they obtained mixed results when colorizing grayscale images using the Places365 dataset. Mis-colorization was a frequent occurrence with images containing high levels of textured details. This leads us to believe that the model has identified these regions as grass since many images in the training set contained leaves or grass in an open field. In addition, this network was not as well-trained as the CIFAR-10 counterpart due to its significant increase in resolution (256×256 versus 32×32) and the size of the dataset (1.8 million versus 50, 000).

2.4 RECOVERING OLD OR DAMAGED IMAGES USING GAN

Harshith, Chandan Kumar and K S Mahesh

The ideology of the team in this paper is that they address the limitations of traditional methods that rely on hand-coding loss functions and to present an approach that leverages GANs for more satisfying results. The colorization Model is a deep learning model that takes a grey scale image as input and produces a coloured image. Around 8000 images of size 256 * 256 were used for training. The models were evaluated using various metrics like precision, recall, f1 score and Fréchet inception distance (FID) for accuracy, precision and other evaluation metrics.

2.5 PalGAN:IMAGE COLORIZATION WITH PALETTE GENERATIVE ADVERSARIAL NETWORKS

Yi Wang, Menghan Xia, Lu Qi, Jing Shao, and Yu Qiao

The team lead by Yi Wang formulated colorization as a palette prediction and assignment problem. Compared with directly learning the pixel-to-pixel mapping from grey to colour as adopted by most learning-based methods, this disentanglement fashion not only brings empirical colorization improvements, but also enables us to manipulate global colour distributions by adjusting or regularizing palette.

2.6 COLORIZING BLACK AND WHITE IMAGES USING CONDITIONAL GAN

T. Sai Srinivas, Vemuri Harshitha, Balabolu Harshitha and Mansa Devi

In this project, They proposed an optimized approach for image colorization using a combination of CGAN and UNet. They first trained a CGAN to generate colour images conditioned on the corresponding grayscale images. Then, they used the generator network of the trained CGAN as an encoder in a U-Net architecture. The decoder network of the U-Net is then trained to generate the colour image from the encoded features. The experimentation's findings demonstrated that, when applied to benchmark datasets, the suggested strategy outperforms current state-of-the-art approaches. The proposed approach produces visually appealing and realistic color images with improved accuracy and efficiency.

2.7 IMAGE -TO-IMAGE TRANSLATION WITH CONDITIONAL ADVERSARIAL NETWORKS

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou and Alexei A. Efros

Phillip Isola and his team investigated conditional adversarial networks as a general-purpose solution to image-to-image translation problems. They concluded that these networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally

would require very different loss formulations. They also demonstrated that this approach is effective at synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images, among other tasks.

2.8 COLORIZATION USING CONVNET AND GAN

Qiwen Fu, Wei-Ting Hsu and Mu-Heng Yang

In this paper they, implemented two models for the task: ConvNet and conditional-GAN and found that GAN can generate better results both quantitatively and qualitatively. They also observed that the quality of the images generated by GAN is in general higher than the images generated by the ConvNet with L2 loss. Images generated by GAN are brighter and clearer. Sometimes even brighter than the ground truth image.

2.9 GAN-BASED IMAGE COLORIZATION FOR SELF-SUPERVISED VISUAL FEATURE LEARNING

Sandra Treneska, Eftim Zdravevski, Ivan Miguel Pires, Petre Lameski and Sonja Gievska

In this paper, the first focus is on image colorization with generative adversarial networks (GANs) because of their ability to generate the most realistic colorization results. Then, via transfer learning, they used this as a proxy task for visual understanding. Particularly, they also proposed to use conditional GANs (CGANs) for image colorization and transfer the gained knowledge to multilabel image classification and semantic segmentation.

This was the first time that GANs have been used for self-supervised feature learning through image colorization. Through extensive experiments they noticed an increase of 5% for the classification task and 2.5% for the segmentation task.

2.10 IMAGE COLORIZATION USING SELF ATTENTION GAN

Amey Narkhede, Rishikesh Mahajan and Priya Pacharane

Amey, Rishikesh and Priya trained the generator and discriminator using Adam as optimizer. The initial learning rate for both generator G and discriminator D are set to 0.0002. The model was implemented in PyTorch framework and trained on Nvidia GPU with 11GB of memory. They observed that after the inflection point the image quality oscillates between best and bad. They have shown that image colorization with self attention GAN comes closer to produce a more plausible and realistic colour photos. They also proposed a new GAN training method to shorten the effective training time.

CHAPTER 3

COLORIZATION - PROBLEM AND STRATEGY

3.1 INTRODUCTION TO COLORIZATION

Exploring the challenge of image colorization, this endeavor has entaileda comprehensive study of deep learning literature dedicated to this task. While colorization initially appeared as a complex problem, subsequent research and engagement with relevant communities led to a more profound understanding of the field. Here, we aim to provide essential knowledge for comprehending the subsequent code implementations.

3.1.1 COMPARING RGB AND L*a*b COLOR SPACES

Image loading results in a 3-dimensional array representing the image's height, width, and color data in the RGB color space. Each pixel is described by three values, denoting the levels of Red, Green, and Blue. Different colors manifest as distinct patterns in their respective color channels. Shifting towardsa particular shade, such as blue, is reflected in higher values in the blue channel. In contrast, the L*a*b color space uses three values to represent each pixel, each with specific meanings. The L channel encodes the pixel's lightness, appearing as a grayscale image when visualized. The *a and *b channels signifythe pixel's green-red and yellow-blue characteristics. This three-channel representation offers more versatility than RGB. It's important to note that the majority of academic papers and open- source repositories in colorization prefer the L*a*b color space for model

training.This choice simplifies the training process as the model is provided with the L channel (representing grayscale) and predicts the other two channels, enablingthe reconstitution of the colored image by concatenating all channels. In contrast, using RGB requires the initial conversion of the image to grayscale before the model predicts three numbers per pixel.

3.1.2 ADDRESSING THE PROBLEM

In recent years, the field of deep learning has witnessed the emergence of various methodologies for image colorization. Notably, the "Colorful Image Colorization" paper approached the challenge as a classification task, recognizing the inherent uncertainty in assigning specific colors to objects within an image. On the other hand, an alternative paper treated the problem asa regression task, incorporating additional adjustments. Each approach carries its advantages and disadvantages. However, in this discussion, we explore a distinct strategy.

3.2 THE STRATEGY

The "pix2pix" framework, officially titled "Image-to-Image Translation with Conditional Adversarial Networks," introduced a highly adaptable approach to resolving various image-to-image challenges within the domain of deep learning. This particular technique capitalizes on two fundamental loss functions: the L1 loss, effectively reframing the problem as a regression task, and the adversarial (GAN) loss, enabling unsupervised problem-solving by assessing the authenticity of generated outputs within the GAN framework, two pivotal components collaborate to address the problem: the generator and the discriminator models. In this context, the generator takes a one-channel grayscale image as input and generates a two-channel image, consisting of channels for *a and *b. Subsequently, the discriminator assesses the authenticity of this newly created three-channel image, formed by concatenating the two generated channels

with the initial grayscale image. To fulfill its learning objectives, the discriminator also encounters authentic three-channel images in the Lab color space, which were not produced by the generator.

The concept of "condition" in this context pertains to the grayscale image shared as input to both the generator and discriminator models within our GAN framework. This input condition serves as the basis for our expectations that both models will consider it during their respective processes. In terms of mathematical representation, we denote the grayscale image as "x," the input noise for the generator as "z," and the desired two-channel output from the generator (which can also represent the two colour channels of a real image) as "y".

Additionally, "G" signifies the generator model, and "D" denotes the discriminator. Refer to figure 3.1 for the formula of loss function. Observe thatthe variable "x" serves as the shared condition introduced to both participants in this framework. However, it's worth noting that we deviate from the conventional practice of feeding the generator a multidimensional vector of random noise, opting for a unique approach. Instead of providing the generatorwith an "n"-dimensional vector of random noise, as one might expect, we incorporate noise through the utilization of dropout layers integrated into the generator's architecture.

$$\begin{aligned}\mathcal{L}_{cGAN}(G, D) =& \mathbb{E}_{x,y}[\log D(x, y)] + \\ & \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]\end{aligned}$$

3.2.1 OPTIMIZATION

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]$$

The initial loss function contributes to the generation of lifelike and authentic colorful images. To enhance model performance and introduce a degree of supervision, we combine this loss function with the L1 Loss, also recognized

as mean absolute error, for comparing predicted and actual colors. Using L1 loss in isolation prompts the model to colorize images, albeit with a tendency towards conservative color choices, such as "gray" or "brown."

In cases of uncertainty regarding the ideal color, the model adopts a cautious approach, favoring the average and employing these colors to minimize the L1 loss, akin to the blurring effect associated with L1 or L2 loss in super-resolutiontasks. The preference for L1 Loss over L2 loss, or mean squared error, mitigatesthe generation of grayish images. The combined loss function can be seen in Figure 3.3 below. **λ** is a coefficient to balance the contribution of the two lossesto the final loss.

The combination function:

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

CHAPTER 4

SYSTEM ARCHITECTURE

4.1 ARCHITECURE

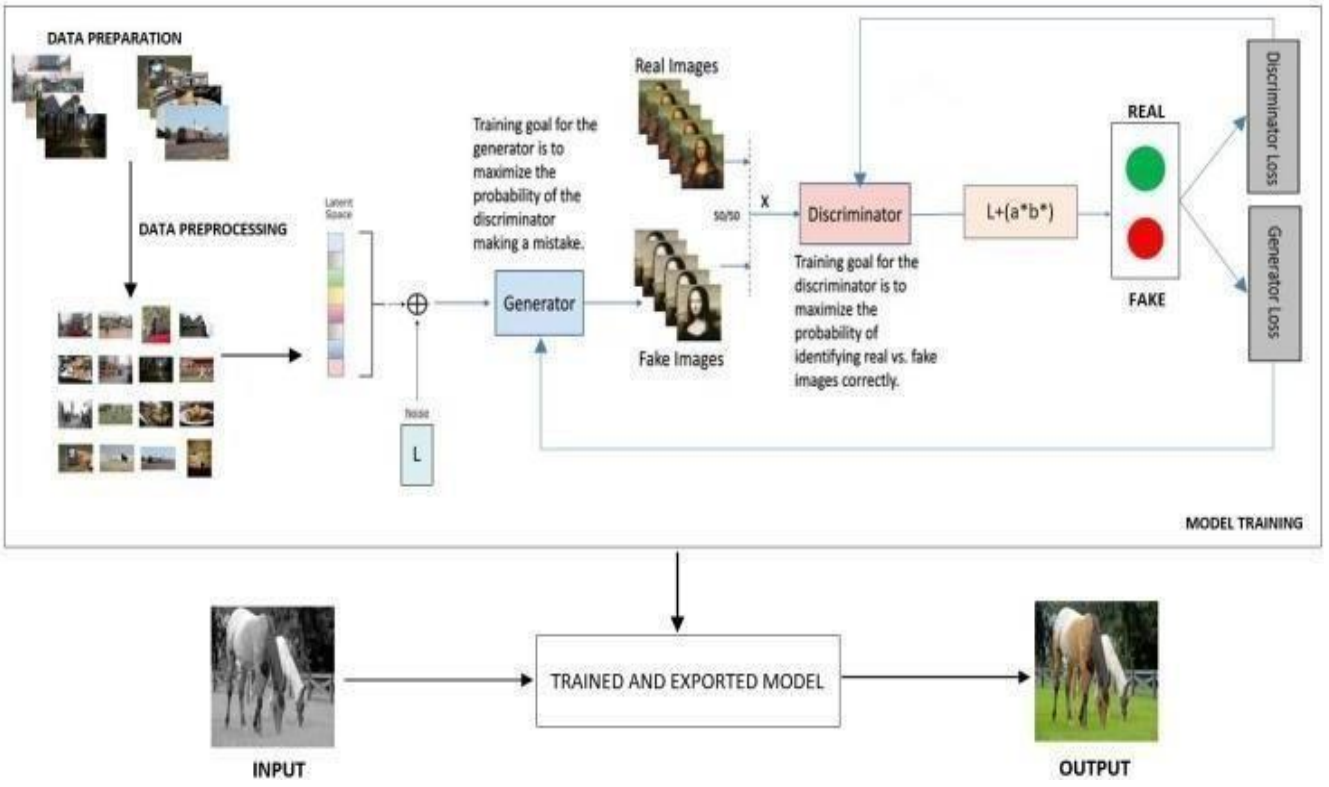

Figure 4.1 Architecture Diagram for GANs

4.2 COMPONENTS AND ITS WORKING

Colorization is a remarkable application of Generative Adversarial Networks (GANs), where the Generator, Discriminator, latent space, noise, loss functions, and fine-tune training all work in tandem to transform grayscale images into realistic, vibrant color images. Let's explore how these elements collaborate in the colorization process:

Generator for Colorization:

The Generator in a colorization GAN is designed to accept grayscale images (1-channel) as input and produce colorized versions (2-channel: a and b channels*)* as output. It operates in the *La*b color space,where the L channel represents the grayscale information, and the *a and *b channels encode color details.

Discriminator's Role:

The Discriminator plays a crucial role in distinguishing the colorized images generated by the GAN from real, color images. It evaluates the authenticity of the images produced by the Generator by comparing them with real color images in the dataset.

Latent Space for Creativity:

The latent space remains integral to introduce variability in the colorization process. Grayscale images are mapped to this latent space, where random noise is added. This noise perturbs the Generator's transformation process, resulting in different colorization outcomes for the same grayscale input. It allows for diversity in colorization.

Noise Injection:

Noise is added to the latent space, and it is introduced into the Generator architecture. The presence of noise in the transformationprocess ensures that the GAN does not produce identical colorizations for the same grayscale input, contributing to diversity among the generated color images.

Loss Functions in Colorization:

Loss functions guide the colorization process. They are employedto evaluate how well the generated colorized images match real color images. While the adversarial loss encourages the Generator to create images that are indistinguishable from real ones, the L1 loss (mean absolute error) is often used

to ensure pixel-wise accuracy incolorization. These loss functions drive the fine-tuning process.

Fine-Tune Training:

The GAN undergoes an iterative training process to improve the quality of colorization. The Generator and Discriminator are refined during training. The Generator aims to create colorized images that can deceive the Discriminator, while the Discriminator becomes more adeptat distinguishing real color images from fake ones. As training progresses, colorizations become increasingly realistic, capturing intricate patterns and vibrant colors.

In summary, a GAN for colorization combines various components to bring grayscale images to life with vivid and realistic colors. The Generator takes grayscale inputs, leverages the latent space and noise to introduce variability, and employs loss functions to guide colorization quality. Through fine-tune training, the Generator refines its ability to create lifelike color images, making it a powerful tool for image colorization tasks.

CHAPTER 5

REQUIREMENTS SPECIFICATION

5.1 HARDWARE REQUIREMENTS

The most common set of requirements defined by any software application is the hardware. The Hardware Requirements of the system are:

Processors	Basic Multi-core processor
Random Access Memory	Minimum 16GB is recommended
GPU (recommended)	NVIDIA GeForce with CUDA

5.2 SOFTWARE REQUIREMENTS

Software requirements deals with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. The Software Requirements of the system are:

Operating Systems	MS Windows 10 or Higher
Python Version	v3.x or above
Development Tools	Jupyter Notebook, Google colab
Libraries	NumPy, Pandas, scikit-image, matplotlib, PIL, Tqdm
Deep Learning Framework	PyTorch v1.0

CHAPTER 6

IMPLEMENTATION MODULES AND RESULTS

6.1 DATASET SELECTION AND EXPLORATION

In our first module we embark on the initial stages of our colorization project, which hinge on the careful selection and exploration of the dataset. Thedataset forms the bedrock of any machine learning project, and in this module,we establish the crucial components required to advance in our colorization task.

Our starting point is the input. We acquire a diverse collection of image data, drawing from a wide array of scenarios and subjects. This dataset is the raw material upon which our colorization model will be trained. We obtain these images from a specified path, and they serve as the foundation upon which the project unfolds. In terms of output, we aim to attain a comprehensive understanding of the dataset itself. This encompasses grasping its overall structure, assessing itssize, and comprehending its composition. A core outcome of this module is theidentification of specific subsets within the dataset. We meticulously distinguish between images designated for training and those set aside for validation.

The implementation process involves gathering the image data, a fundamental asset for our colorization model. We engage in a process of selecting a subset of 10,000 images, drawn randomly from the expansive COCO dataset. These images represent a wide spectrum of scenes, objects, andcolors, a crucial prerequisite for training a robust colorization model. Randomization plays a pivotal role in this endeavor, ensuring an equitable distribution of images across the

selected dataset. This approach fosters diversity, covering a broad spectrum of colors and scenes. With this diverse dataset in hand, we are poised to proceed with the subsequent phases of our colorization project.

Furthermore, we carefully delineate our dataset into distinct subsets: the training and validation subsets. The training subset encompasses the first 8,000 images, while the remaining 2,000 images are designated for validation. This division facilitates model evaluation and ensures the model's performance extends beyond the images used during training.

Another valuable facet of this module is the visual exploration it provides. We are presented with visual previews of a subset of randomly selected images from our dataset. These previews grant us a profound understanding of the dataset's content, showcasing the variety of scenes and objects encompassed within. These glimpses into the dataset's richness furnishus with critical context for the upcoming stages of our colorization journey.

6.2 DATA PREPARATION AND DATALOADER CREATION

In this segment, we delve into the essential data preparation and data loader creation processes, which are crucial for our colorization task. The primary objective here is to ensure that the dataset is properly formatted and ready for use in the subsequent stages. First, we emphasize the image size, denoted as ‘SIZE’, which is set to 256. This standardization ensures consistentinput dimensions for our model.

For the creation of data loaders, we define a custom ‘Colorization Dataset’ class. This class plays a pivotal role in dataset management, as it handles the loading and preprocessing of image data for both training and validation. When the split is designated as ‘train’, the dataset undergoes additional preprocessing steps.

Images are resized to match the predefined ‘SIZE’, and random horizontal flips are applied for a bit of data augmentation. On the other hand, for the ‘val’ split, images are resized to ensure uniform dimensionswithout augmentation.

The actual data preprocessing involves several steps. Each image in our dataset is opened, converted to the RGB format, and resized according to the split type. These resized images are then converted to the LAB color space using the ‘rgb2lab’ function. The LAB color space separates an image into its Lightness (L) and color channels (a and b), providing a more structuredrepresentation for our colorization task. Once in the LAB color space, the imagedata is normalized, and the Lightness (L) channel is adjusted to fall within the range of -1 to 1, while the color channels (a and b) are adjusted to range from -1 to 1 as well.

To facilitate efficient data management and batching, a helpful function called ‘make_dataloaders’ is introduced. This function enables the creation of dataloaders, allowing us to load and preprocess data seamlessly. These dataloaders are configured with parameters such as batch size, the number of workers for parallel processing, and whether to pin memory for faster data transfer to GPU. They play a pivotal role in the training process, ensuring that the model can efficiently access and process the data during training.

In the provided code, these dataloaders are utilized for both the training an validation datasets. They act as essential conduits, enabling the flow of datato the model during the training and evaluation phases. Furthermore, the utility of dataloaders is demonstrated by fetching a batch of data from the training dataloader and confirming the dimensions of the Lightness and color channels within the batch.

6.3 GENERATOR USING UNet ARCHITECTURE

The UNet architecture plays a pivotal role in the colorization process. It comprises a series of interconnected blocks that facilitate the colorization of grayscale images. These blocks are instrumental in generating the color information for each pixel based on the input grayscale image. The UNet architecture consists of two main components: the UnetBlockand the Unet model. The UnetBlock is a fundamental building block within theUNet architecture. It is a flexible module that can be configured for various purposes, including serving as the innermost or outermost component of the network. This adaptability allows the UNet architecture to efficiently capture and transform image features.

The UnetBlock's inner workings involve a series of convolutional operations and non-linear activation functions. It performs both downsampling and upsampling operations using convolutional layers. While downsampling involves reducing the spatial dimensions of the image, upsampling increases it. This mechanism allows for the extraction of essential image features at multiplescales. The UNet model, on the other hand, orchestrates the complete architecture. It defines the overall structure and connectivity of the UNet blocks. This model facilitates the flow of information and gradients across the network. The architecture consists of a series of UNetBlocks, each connected to the previous one. These connections enable the UNet to progressively refine and generate the colorization information for the grayscale input.

Throughout the UNet architecture, convolutional layers are used for feature extraction and transformation. Additionally, normalization techniques like batch normalization are applied to ensure stable and efficient training. Activation functions, such as Leaky ReLU and ReLU, introduce non-linearity,allowing the network to capture complex relationships in the data.

One crucial aspect of the UNet architecture is its ability to handle different

levels of image detail. As the network progresses through the UNetBlocks, it can capture both fine and coarse features, enabling it to produce high-quality colorizations. Furthermore, the UNet architecture incorporates dropout layers to prevent overfitting during training. These dropout layers introduce regularization, improving the network's generalization capabilities.

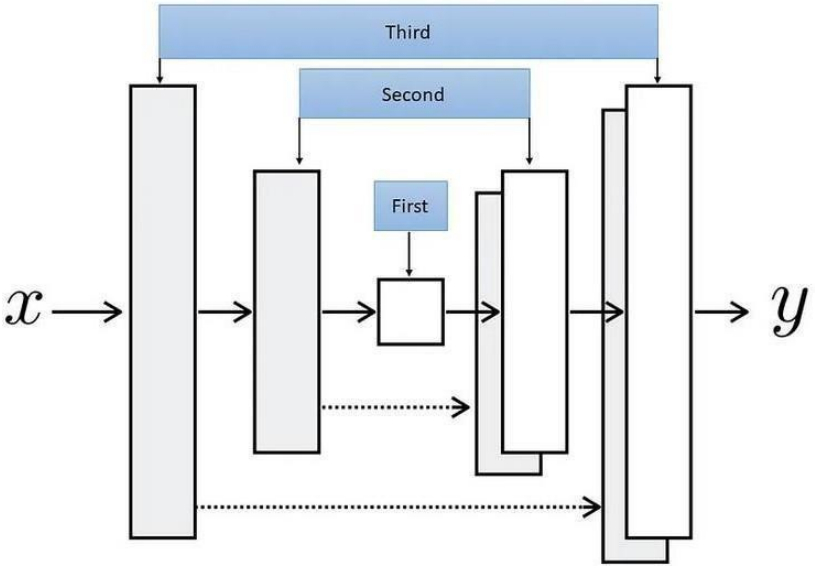

Figure 6.1 UNet from the paper Image-to-Image translation with Conditional Adversarial Networks

6.4 DISCRIMINATOR USING PatchGAN

The discriminator, known as a PatchGAN, plays a crucial role in the adversarial learning process of the colorization model. This component is responsible for distinguishing between real and generated colorized images. The discriminator is designed as a neural network that assesses localized patches within an image to provide feedback to the generator.

The PatchDiscriminator is defined as a neural network module. It acceptsan input image or feature map and processes it to make binary judgments aboutthe realism of the image patches. To achieve this, the discriminator is equippedwith

multiple convolutional layers, each with specific characteristics to capturerelevant image features.

The architecture of the PatchDiscriminator consists of several layers, starting with a convolutional layer that takes the input image. Each subsequent layer includes convolutional operations, and depending on the location within the network, may involve normalization and activation functions. The convolutional layers aim to extract information that will assist in determining the authenticity of image patches.One notable feature of the PatchDiscriminatoris the application of a Leaky ReLU activation function. This activation functionintroduces a non-linearity that helps the discriminator detect subtle differencesbetween real and generated patches. Additionally, the use of Leaky ReLU allows the network to handle complex variations in the images. The network's architecture also includes batch normalization, a techniquethat stabilizes training by normalizing the inputs to each layer. However, it's important to note that the final layer of the PatchDiscriminator does not employbatch normalization or an activation function. This design choice ensures that the last layer produces a raw output that can be used for adversarial learning.

During the training process, the PatchDiscriminator evaluates local patches of both real and generated images, enabling it to provide detailed feedback about the realism of these patches. The discriminator's goal is to identify discrepancies between the real and generated patches and guide the generator to produce more realistic colorizations. In practice, the PatchDiscriminator is employed as part of the GAN (Generative Adversarial Network) framework, where it collaborates with the generator. It helps the generator improve its colorization quality by learning from the discriminator's feedback.

Overall, the PatchDiscriminator is a critical component in the adversarial learning process, allowing the colorization model to produce more convincing and

realistic colorized images.

6.5 GAN LOSS

The GAN (Generative Adversarial Network) Loss, a crucial component of the colorization model, is responsible for quantifying the performance of the adversarial learning process. This loss function plays a fundamental role in guiding the generator and the discriminator to achieve their objectives effectively.

The GAN Loss is implemented as a neural network module called GANLoss. It is designed to support two common GAN modes: "vanilla" GANand "LSGAN" (Least Squares GAN). The choice between these modes allowsfor flexibility in adversarial learning, catering to different tasks and objectives.In the context of adversarial learning, the discriminator and the generator engage in a competitive game. The generator aims to produce colorized imagesthat are indistinguishable from real ones, while the discriminator attempts to correctly classify whether an image is real or generated. The GAN Loss quantifies the degree to which the discriminator can differentiate between real and generated images.

The core of the GAN Loss is the binary cross-entropy with logits loss (BCEWithLogitsLoss) in the "vanilla" GAN mode. In this mode, the loss function computes the binary classification loss based on the logits produced by the discriminator. The BCEWithLogitsLoss measures the difference between the predicted probabilities and the target labels, where real images arelabeled with 1.0 (indicating "real") and generated images with 0.0 (indicating "fake").

In contrast, the "LSGAN" mode uses the mean squared error loss (MSELoss). This mode has the goal of minimizing the squared differences between the discriminator's predictions and the target labels. "LSGAN" is oftenfavored for its ability to produce more stable and continuous gradients during training.

The GAN Loss provides the flexibility to adapt to the needs of the colorization model by allowing users to choose the appropriate GAN mode. Regardless of the mode selected, the GAN Loss contributes to the adversarial learning process by penalizing the generator for producing colorizations that can be easily distinguished from real images.

Throughout the training process, the GAN Loss is applied during each iteration, guiding the generator to produce colorizations that are convincing and realistic, ultimately improving the overall quality of the model's output. The GAN Loss plays a crucial role in enhancing the colorization model's ability to generate colorized images that closely resemble real ones, making it a pivotal component in the success of the model.

6.6 TRAINING THE MODEL

The training phase of the colorization model is a crucial step in achievingits primary objective of adding color to grayscale images. This module focuseson the training process, which involves optimizing the model's parameters andupdating the loss values. The goal is to ensure that the generator produces high- quality colorized images that closely resemble real ones. The training process is characterized by several key components, each contributing to the overall success of the model. Let us dive into these components:

Data Preparation

Before training commences, the dataset must be prepared. The dataset consists of both grayscale and corresponding color images. Grayscale images are used as input, while color images serve as the ground truth during training. The dataset is organized into training and validation sets to monitor the model's performance and avoid overfitting.

Training Batches

The training data is divided into batches, each containing a specified number of samples. These batches are fed into the model during training to facilitate optimization. This process ensures that the model learns from a variety of input images.

Loss Meter Initialization

At the beginning of each training epoch, loss meters are initialized.These loss meters are essential for tracking the performance of the modelduring training. They help monitor how well the generator and discriminator are doing in terms of adversarial and reconstruction tasks.

Epoch Iteration

The training process is organized into epochs, with each epoch consisting of multiple iterations. During each iteration, a batch of trainingdata is processed by the model.

Model Setup and Optimization

Within each iteration, the model is set up with the input data. Thegenerator takes grayscale images as input and generates colorized predictions. The discriminator evaluates the generated images for authenticity.

Updating Losses

After model evaluation, the loss values are updated. Different types of losses are calculated, such as the adversarial loss, perceptual loss, and reconstruction loss. These losses are critical for guiding the model in producing high-quality

colorizations.

Logging Results

Periodically, the training process logs results to monitor progress. This includes printing out relevant information, such as the current epoch, iteration, and the values of different loss components. It helps in assessing the model's performance and provides insights into its learning progress.

Visualization

A visual inspection of model outputs is performed at fixed intervals. This visual feedback aids in understanding how well the model is doing in generating colorized images.

Epoch Completion

The process repeats for the specified number of epochs. Each epoch fine-tunes the model parameters, helping it generate increasingly realistic and accurate colorizations. The number of training epochs is a hyper parameter that can be adjusted based on the desired level of modelperformance. Training for more epochs allows the model to learn better,but it must be balanced.

6.7 UTILITY FUNCTIONS

Utility functions are essential components of the colorization model's codebase, providing critical functionality and aiding in the research process. These functions are meticulously designed to handle various tasks, from tracking and logging training statistics to visualizing model outputs. These are just some utility functions to log the losses of our network and also visualize the results during training:

Average Meter

The `AverageMeter` class is a versatile utility for tracking and computing the average of values. During training and evaluation, it keepsa record of statistics and simplifies the process of aggregating andsummarizing results.

Create Loss Meters

The `create_loss_meters` function initializes a set of loss meters, which are crucial for monitoring the performance of the colorization model. These meters are responsible for recording various types of losses, including discriminator and generator losses.

Update Losses

The `update_losses` function enables real-time updates of loss meters. As training iterations progress, this function ensures that the lossmeters accurately reflect the accumulated losses, aiding researchers in assessing the model's performance.

Lab to RGB

`Lab to RGB` is a vital function responsible for converting color channels, specifically L (luminance) and ab (chrominance), back into RGB images. This process is essential for visualizing and interpreting thecolorized outputs of the model.

Visualize

The `visualize` function is a dynamic tool for visualizing model results. It allows researchers to observe and assess the quality of colorized images and compare them with ground truth images. This function enhances the interpretability of model performance.

Log Results

Logging is a critical aspect of research, and the `log_results` function fulfills this role by printing the average values of loss meters tothe console. This real-time feedback aids in monitoring training and evaluating the model's convergence. These utility functions are the backbone of the research framework, offering efficiency and transparency in managing the training process, monitoring model performance, and ensuring the robustness of the colorization model.

6.8 RESULTS

TRAINING PROCESS AND LOSS METRICS

Iteration 200/500:

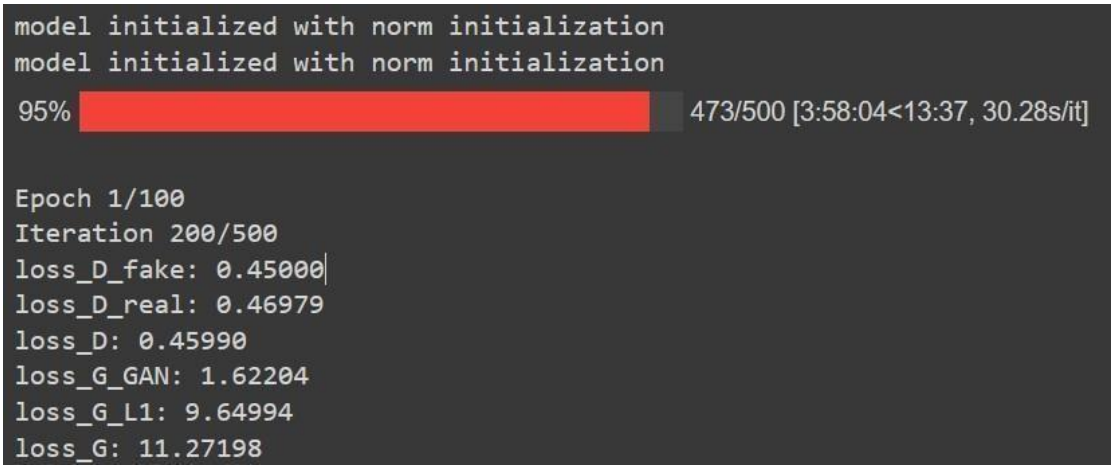

Figure 6.2 Metrics Visualization

In the provided output, we have two main sections:

Model Initialization:

"model initialized with norm initialization" is a message indicatingthat the model has been initialized with normalized weights. This typically means that the model's weights have been set using a normalization technique to ensure stable and efficient training. Proper weight initialization is crucial for the convergence and

performance of neural networks.

Training Progress:

- "95%" suggests the completion progress of a training process,specifically, 95% of the process is complete.
- "473/500" indicates that the training process has iterated throught
- 473 out of 500 total iterations.
- "[3:58:04<13:37, 30.28s/it]" shows the training time progress.

Following this, the output provides details about the training process for a specific epoch (Epoch 1/100) and iteration (Iteration 200/500):

"loss_D_fake," "loss_D_real," "loss_D," "loss_G_GAN," "loss_G_L1," and "loss_G" are various loss metrics associated with training a GAN (Generative Adversarial Network). The numbers associated with these loss metrics indicate the valueof each loss at the given training stage. For example, "loss_D_fake: 0.45000" suggests the loss value for the discriminator when evaluating fake data.

OUTPUT

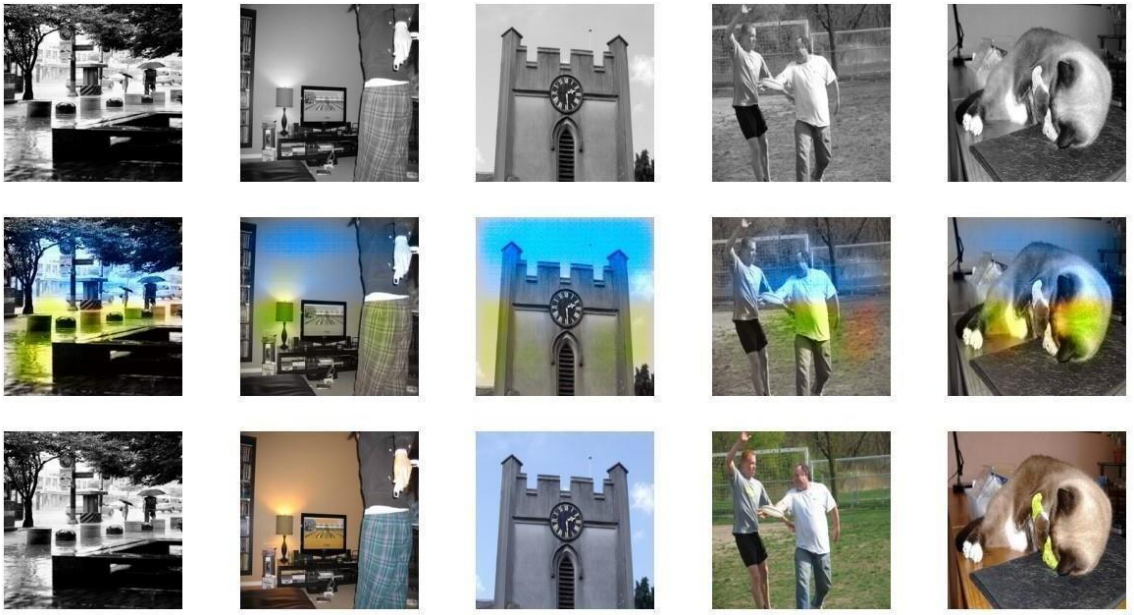

Figure 6.3 Output after 200 iterations

Iteration 400/500:

```
Epoch 1/100
Iteration 400/500
loss_D_fake: 0.47620
loss_D_real: 0.49065
loss_D: 0.48342
loss_G_GAN: 1.55394
loss_G_L1: 9.96936
loss_G: 11.52330
```

Figure 6.4 Metrics of a later stage

Figure 6.5 Output after 400 iterations

VISUALIZATION IN EARLIER STAGES FOR ANALYSIS

```
_, axes = plt.subplots(4, 4, figsize=(10, 10))
for ax, img_path in zip(axes.flatten(), train_paths):
    ax.imshow(Image.open(img_path))
    ax.axis("off")
```

Figure 6.6 Creates a 4x4 grid for visualization

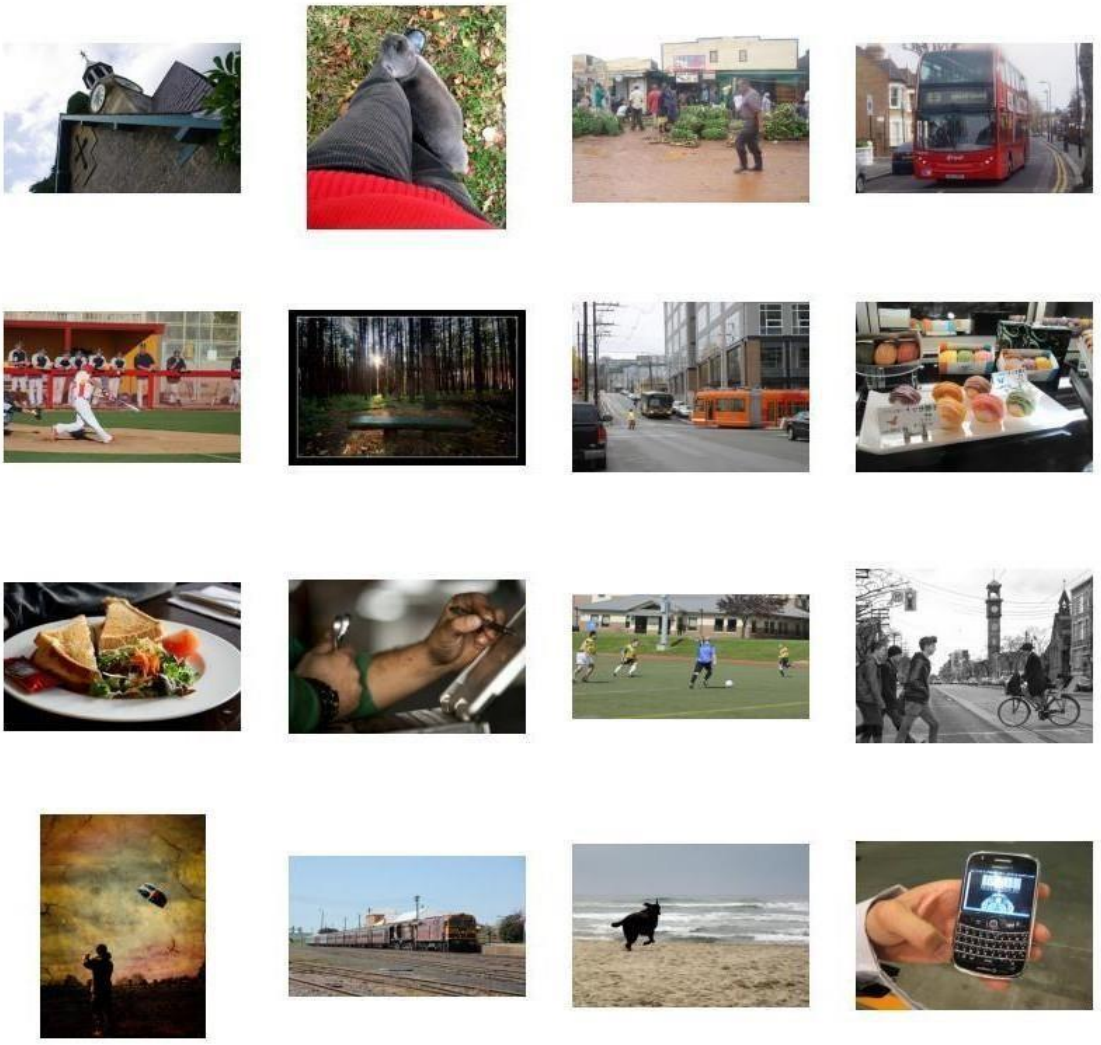

Figure 6.7 4x4 Grid

CHAPTER 7

CONCLUSION AND FUTURE WORKS

In this thesis, we have embarked on a comprehensive journey into the world of Generative Adversarial Networks (GANs) and their application in image colorization. GANs represent a transformative paradigm in the realm of image generation, offering the potential to create vivid and realistic content that aligns with the distribution of the training data. The fundamental architecture of GANs, featuring a Generator and a Discriminator engaged in an adversarial interplay, serves as the cornerstone of this technology.

Our exploration extended to the profound concept of the latent space, a vital element that empowers the GAN to produce diverse and unique images, making them versatile and adaptable to various creative domains.

Moving forward, we delved into the intricacies of image colorization, a process that bestows grayscale images with vibrant colors. We underscored the role of loss functions, notably the L1 loss, in guiding the GAN to create colorized images that are visually appealing and exhibit realism.

Our journey further led us into the core of the colorization process, dissecting the architecture of the Generator, which embodies a UNet-like structure. This design effectively captures features and spatial relationships within images, enhancing the colorization process's precision. The Discriminator, crafted as a PatchGAN, emerged as a discerning classifier of image patches, distinguishing between real and generated content.

Delving deeper into the GAN framework, we scrutinized the GAN loss, encompassing modes like "vanilla" and "lsgan." These loss functions serve as a compass during training, driving the GAN to generate images that confound the Discriminator.

The practical implementation of the training process was unveiled, highlighting the utilization of utility functions, loss meters, and visualization techniques. This phase empowers the GAN to learn and refine its colorization capabilities over time, setting the stage for the creation of compelling colorized images.

In our quest to harness GANs for image colorization, we encountered intriguing challenges and nuances, such as mode collapse and color distribution. However, by harnessing the capabilities of conditional GANs, we gained the ability to generate images of specific types or classes.

REFERENCES

1. A. Radford, L. Metz, and S. Chintala. (2015), “Unsupervised representation learning with deep convolutional generative adversarial networks”, arXiv preprintarXiv:1511.06434.

2. Bolei Zhou. (2021), “Interpreting Generative Adversarial Networks for Interactive Image Generation”, The Chinese University of Hong Kong, License: CC BY 4.0.

3. C. Li and M. Wand. (2016), “Precomputed real-time texture synthesis with markovian generative adversarial networks”, ECCV.

4. Cheng, Zezhou, Qingxiong Yang, and Bin Sheng. (2016), "Deep Colorization." European Conference on Computer Vision (ECCV).

5. Harshith J, Chandan Kumar M. (2021), “ Recovering Old or Damaged Images using GAN”, International Journal of Advanced Research in Computer and Communication Engineering Vol. 10, Issue 5, DOI 10.17148/IJARCCE.2021.105173.

6. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu,D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio.(2016), “Generative adversarial nets. In Advances inneural information processing systems”, pages 2672–2680.

7. Ironi, R., Cohen-Or, D., Lischinski. (2015),“Colorization by example Rendering techniques”,201–210.

8. J. Zhao, M. Mathieu, and Y. LeCun. (2016), “Energy-based generative adversarial network”, arXiv preprint arXiv:1609.03126.

9. K. Hong, J. Li, W. Li, C. Yang, M. Zhang, Y. Wang, and Q. Liu. (2018), “Joint intensity-gradient guided generative modeling for colorization,” arXivpreprint arXiv:2012.14130.

10. K. Simonyan and A. Zisserman (2017), “Very deep convolutional networks for large-scale image recognition” , Conference Track Proceedings.

11. K.Nazeri and E. Ng. M. (2020),“Image Colorization with Generative Adversarial Networks” , University of Ontario Institute of Technology Ontario, Canada.

12. Kangning du, Changtong Liu. (2021), “Double-Channel Guided Generative Adversarial Network for Image Colorization”,IEEE Access PP(99):1-1 DOI:11.09/Access.2021.3055575.

13. Karras, T., Laine, S., Aila, T. (2018) ,“A style-based generator architecture for generative adversarial networks” arXiv preprint arXiv:1812.04948.

14. Ledig C, Theis L, Huszár F, Caballero J, Cunningham A, Acosta A, Aitken A, Tejani A, Totz J, Wang Z, et al. (2017) Photo-realistic single image super-resolution using a generative adversarial network.; pp. 4681–4690.

15. L. Metz, and S. Chintala.(2022), “Unsupervised representation learning with deep convolutional generative adversarial networks,” arXiv preprint arXiv:1511.06434.

16. Liu, Yijun, et al. (2017) ,"Colorful Image Colorization with Sketched Gradients via Neural Network Optimization". Computer Vision and Pattern Recognition (CVPR), IEEE Computer Applications.

17. M. Mirza and S. Osindero, (2017) ,“Conditional generative adversarial nets,” CoRR, vol. abs/1411.1784.

18. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. (2016), “Image-to-image translation with conditional adversarial networks,” arXiv preprintarXiv:1611.07004.

19. Raj Kumar Gupta, Alex Yong-Sang Chia, Deepu Rajan, Ee Sin Ng, and Huang Zhiyong. (2012), “Image colorization using similar images”, Proceedings of the 20th ACM International Conference on Multimedia.

20. Reinhard E., Ashikhmin M., Gooch B., and Shirley. (2001) ,“Color transfer between images”, IEEE Computer Graphics and Applications.

21. S. Iizuka, E. Simo-Serra, and H. Ishikawa. (2016) , “Let there be color!: joint end to- end learning of global and local image priors for automatic image colorizationwith simultaneous classification,” ACM Trans. Graph., vol. 35, no. 4, pp. 110:1–110:11.

22. T. Sai Srinivas, Vemuri Harshitha. (2023), “Colorizing black and white images using conditional GAN” , International Research Journal of Modernization in Engineering Technology and Science , Volume:05 ,Issue:04, e- ISSN:2582-5208.

23. T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. (2016), “Improved techniques for training gans”, CoRR, abs/1606.03498.

24. Takeru Miyato and Toshiki Kataoka and Masanori Koyama and Yuichi Yoshida. (2018), “Spectral Normalization for Generative Adversarial Networks”, International Conference on Learning Representations.

25. X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. (2017), “Least squares generative adversarial networks,” in IEEE International Conference onComputer Vision, pp. 2813–2821.

26. Y. Zhao, L. Po, K. Cheung, W. Y. Yu, and Y. A. U. Rehman. (2020), “SCGAN: saliency map-guided colorization with generative adversarial network,” CoRR,vol. abs/2011.11377.

27. Yi Wang, Menghan Xia, Lu Qi, Jing Shao, and Yu Qiao. (2021), “PalGAN: Image Colorization with Palette Generative Adversarial Networks”, Shanghai AI Laboratory, Tencent AI Lab and Sense Time Research.

28. Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. (2014), “Image quality assessment: from error visibility to structural similarity”. IEEE Transactions on Image Processing, 13(4):600–612.

29. Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. (2018), “Unet++: A nestedu-net architecture for medical image segmentation” , Proceedings, pp. 3–11.

30. Zohaib Mushtaq, Shun-Feng Su. (2020), “Environmental sound classification using a regularized deep convolutional neural network with data augmentation”, Applied Acoustics, Vol. 67, p. 107389, DOI: 10.1016.