A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN

Mekonnen Bayisa Dame

A Thesis Submitted to the Department of Computer Science and Engineering
College of Electrical Engineering and Computing

September, 2025
Adama, Ethiopia

A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN

Mekonnen Bayisa Dame

Advisor: Mesfin Abebe (Ph.D.)

A Thesis Submitted to the Department of Computer Science and Engineering
College of Electrical Engineering and Computing

Office of Graduate Studies
Adama Science and Technology University

September, 2025
Adama, Ethiopia

DECLARATION

I hereby declare that this Master's Thesis entitled “A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN” is my work and has not been submitted to any University for a similar purpose. All sources of materials that are used for this thesis have been duly acknowledged through citation.

Mekonnen Bayisa Dame		
Name of Student	Signature	Date

RECOMMENDATION OF ADVISORS

I, the Major Advisor of this thesis, hereby certify that I have read the revised version of the thesis entitled **"A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN"** prepared under my guidanceby **Mekonnen Bayisa Dame**. Therefore, I recommend the submission of the thesis to the department for further review and evaluation.

Chairperson's	Signature	Date
Reviewer 1	Signature	Date

APPROVAL SHEET

I hereby certify that the recommendations and suggestions given by the thesis review committee are appropriately incorporated into the final thesis entitled **"A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN"** by **Mekonnen Bayisa Dame**.

_____	_____	_____
Chairperson's	Signature	Date

_____	_____	_____
Reviewer 1	Signature	Date

APPROVAL OF THE BOARD OF REVIEWERS

We, the undersigned, members of the Board of Receivers of the thesis open defense by **Mekonnen Bayisa Dame**, have read and evaluated the thesis entitled **"A Hybrid Deep Learning Approach for Color Face Generation Using Mask R-CNN and GAN"** and assessed the understanding of the candidate about the thesis. This is, therefore, to certify that the thesis is accepted for partial fulfillment of the requirements of the degree of Master of Science in Computer Science and Engineering.

Chairperson	Signature	Date
Reviewer 1	Signature	Date

Final approval and acceptance of the thesis is contingent upon submissionof its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and College Graduate Committee (CGC).

Department Head	Signature	Date
College Dean	Signature	Date
Office of Postgraduate Studies, Dean	Signature	Date

ACKNOWLEDGEMENTS

First and foremost, I would like to extend my heartfelt gratitude to Almighty God for His infinite mercy and unwavering guidance throughout the entire journey of my research, from the initial stages to its successful completion. Without His blessings, this work would not have been possible.
I am sincerely thankful to my advisor, Dr. Mesfin Abebe (Ph.D.), for his invaluable guidance, mentorship, and encouragement. His insightful feedback and thoughtful recommendations have been instrumental in shaping the direction and quality of this research.

I also wish to express my deep appreciation to all members of the Data Science, AI, Computer Vision, and Computer Networks SIG. Their continuous support, collaborative spirit, and dedication have created an enabling academic environment that has significantly enriched my postgraduate studies and research progress.
Finally, I am profoundly grateful to my family and friends for their unwavering support, encouragement, and prayers throughout this endeavor.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SAMPLE CODE

LIST OF ACRONYMS

AEGAN	Auto-Embedding Generative Adversarial Network
CGAN	Conditional Generative Adversarial Network
CNN	Convolutional Neural Networks
CV	Computer Vision
DCGAN	Deep Convolutional Generative Adversarial Network
DNN	Deep Neural Network
FCGAN	Fully Connected Generative Adversarial Network
FID	Fréchet Inception Distance
FPN	Feature Pyramid Network
GAN	Generative Adversarial Network
GF	Generated Face Image
GPU	Graphics Processing Unit
GT	Ground Truth Face Image
HVS	Human Visual System
I2I	Image-to-Image Ttranslation
IOU	Intersection over Union
Mask-RCNN	Mask Region-based Convolutional Neural Network
ML	Machine Learning
NLP	Natural Language Processing
PSNR	Peak-to-Noise Ratio
ReLU	Rectified Linear
RPN	Region Proposal Network
SAGAN	Self-Attention Generative Adversarial Network
SSIM	Structural Similarity Index Measure
Tanh	Hyperbolic Tangent Activation Function
VAE	Variational AutoEncoder

ABSTRACT

Generating realistic, high-resolution color face images remains a challenging task in computer vision because it requires both a precise structural representation of facial components and fine-grained texture generation. Existing GAN-based techniques tend to fail in preserving semantic consistency between facial areas and generating high-quality textures, thus resulting in unnatural or blurry images. To solve these issues, this research presents a hybrid deep learning approach that combines Mask R-CNN for semantic facial region segmentation and a GAN-based generator for realistic face image synthesis. Mask R-CNN is used to precisely segment dominant facial features, including eyes, nose, and mouth, which are then employed to guide an attention-enhanced U-Net generator. The generator uses self-attention modules to model long-range spatial dependencies and channel attention to dynamically highlight informative feature channels, thereby preserving local and global image details. A multiscale PatchGAN discriminator enforces image realism at various scales, while training process employs a combination of pixel-wise, perceptual, feature-matching, and structural similarity losses to enhance overall image realism. The introduced method is compared on the CelebA-HQ dataset, with PSNR of 28.33, SSIM of 0.9207, and FID of 21.77, outperforming baseline models and state-of-the-art methods. In addition, the employment of semantic guidance and attention mechanisms enables the model to generalize well even with small or diverse datasets, making it practical for real-world usage scenarios. The results show the framework's promise for high-fidelity facial synthesis in virtual reality, digital media, and other computer vision applications.

Keywords: Color Face Synthesis, GAN, Mask R-CNN, Facial Features Segmentation, and Hybrid Deep Learning

CHAPTER ONE

1. INTRODUCTION

1.1 Background of the Study

The creation of generative adversarial networks (GANs) has boosted the synthesis of color face images at a much faster speed, and their applications span virtual reality, face recognition, movie production, security and among others. GANs, first introduced by Goodfellow et al. in 2014, radically transformed artificial intelligence as they opened the doors toward generating fictional yet believable images in an adversarial learning process (Goodfellow et al., 2014). However, generating photorealistic, high-resolution face images with high quality, superior detail, and wide coverage across ethnicities and skin toners remains unthinkable.

Older GANs, while revolutionary, typically suffer from issues such as low detail, low color fidelity, and a lack of diversity. Because they rely on extensive datasets, they tend to produce blurred outputs when generating high-resolution images. As a result, the majority of models fail to produce refined textures, shading, and fine face details essential for realism. Additionally, narrow variability in sample datasets tends to yield outputs with dominating narrow skin color and face traits, thus reducing their global representativeness (Radford, 2015).

The limitations of these, though, prompted the creation of deep GAN models, notably the StyleGAN and the Auto-Embedding GAN (AEGAN). The former had introduced a vector of "style" that provided high-accuracy control in lighting, pose, and expression (Karras, Laine, and Aila., 2019). Although it absolutely excelled in the situation of face synthesis, however, it is still underperforming in the situation of the multi-ethnic face due to biases in the set. The AEGAN, meanwhile, has an auto encoder that seeks to identify the general pattern through the assistance of a denoiser that sharpens details, thereby elevating sharpness as well as structure accuracy (Guo et al., 2019). Its purpose remains, however, constrained by the specific characteristics of the set that it is being gleaned from and thus inhibit generalization as well as variety.

These limitations place a requirement for a solution that can provide high detail enhancement alongside region-variant control in synthesizing faces. Combining Mask R-CNN with GANs provides an interesting solution (Chang et al., 2024). Mask R-CNN is better suited to object segmentation and object detection, whereby the regions of the face, i.e., nose, eyes, and mouth, can be accurately manipulated (He et al., 2017). In combination with GANs, the combination

provides improved texture detail, color accuracy, and structure similarity, accompanied by varied representation in skin shades and race.

The present work puts forth a hybrid model that borrows the strengths of the GAN and the Mask R-CNN in order to generate high-quality, photorealistic color face pictures. The GAN part achieves the big-level realism as well as the structural integrity, whereas the region-level detailing is accomplished by the Mask R-CNN part. The two, in combination, are proposed as the solution to two very important issues in the production of face images, that being creating photorealistic output as well as representing various populations.

1.2 Motivation of the Study

The developments in information technology, computer vision, and artificial intelligence during the recent past have raised fears of misuse of facial photographs, such as illegal sharing and exploitative manipulation. The present processes of colorizing historic black-and-white photographs and early photographs of pre-fame celebrities are still not satisfactory. Likewise, judicial forensic confirmation reports in injury, disability, and death cases have also been criticized for translating into disproportionately low award values. To address such challenges, this study suggests a color face image generation model that is capable of supporting judicial investigation requirements. An open, large-scale face color image set was employed to improve the training of a convolutional neural network-based model for colorization. In contrast to existing state-of-the-art deep learning methods based on less informative data, the proposed approach yields more realistic and close appearances.

1.3 Problem Statement of the Study

The construction of realistic, colored facial images is a formidable challenge in the field of computer vision, primarily due to the issue of achieving both color fidelity and structural accuracy. Though GANs have shown great promise in image construction (Goodfellow et al., 2014), such networks are often at a disadvantage in developing high-resolution facial features with proper detail in features such as eye morphology, pigmentation in the epidermis, as well as other fine characteristics necessary for obtaining an image that reflects a real human face (Karras, Laine, and Aila, 2019). Though conventional Generative Adversarial Networks are capable of generating a large number of images, they are otherwise disadvantaged by a lack of fine details, suffering several issues, including artifacts, uneven color distribution along blurred high-frequency regions

(Karras et al., 2020). Moreover, an effective definition for specific delineation in faces is essential so that constructed faces are a realistic representation of proper human facial expressions along with physiognomy. State-of-the-art models are otherwise at a disadvantage in enabling isolation and fine-tuning specific regions in a face, thanks to their insufficient handling of both structural integrity and color intricacy. There is a proven success in object segmentation tasks for Mask R-CNN in its ability to clearly define a target region in an image (He et al., 2017). Today's challenge is finding a method in which such segmentation potency can be translated into a GAN-based generative model in order for it to improve in constructing facial images. Thus, there is a need for a hybrid approach in maximizing the generative capability in a GAN while taking advantage of a fine segmentation capability in a Mask R-CNN so as to construct detailed, realistic, colored facial images. Overcoming such problems would further open up avenues for applications for constructing synthetic faces for entertainment, digital identity, and privacy-driven applications.

1.4 Research Questions

This study sought to address the following research questions:

RQ1: How does integrating Mask R-CNN as a semantic feature extractor impact the quality of color face generation?

RQ2: What techniques are applied within the GAN model to improve the quality of generated color faces?

RQ3: In terms of image quality and realism, how does the proposed hybrid approach perform compared to standalone GAN models?

1.5 Objectives of the Study

1.5.1 General Objective

The general objective of this research is to create a hybrid Mask R-CNN and GAN model that can produce high-quality, realistic color face images with precise representation of facial features.

1.5.2 Specific Objectives

The following specific objectives were drawn to meet the overall objective of the research:

- Utilize facial image datasets from public sources
- Preprocess the dataset obtained by normalization, resizing, and aligning face images to

make the data ready for training the models.

- Utilize a pre-trained Mask R-CNN model for accurate facial region segmentation
- Create and train a GAN model for generating realistic and high-quality color face images
- Hybridize Mask R-CNN with a GAN model in a way that the model learns to give more attention to valuable facial region such as eyes, nose, and mouth.
- Train the hybrid model to generate realistic and high-resolution color face images given condition on segmented mask regions.

1.6 Significance of the Study

This work is advancing one of the frontiers of computer vision issues: generating photorealistic-colored high-resolution faces with good facial structures and consistent color. With the integration of Mask R-CNN and GANs, this hybrid system presents a solution that not only generates photorealistic faces but also maintains rich structural details through region-adaptive importance assignment. This boost is its value in gold to use cases needing realistic and accurate face models, be it facial recognition, augmented reality, animation, or medical imaging. This project aims to advance spatial accuracy and image quality in general and popularizes synthetic dataset generation to fill data gaps for deep learning-based applications. The model enables high-definition face photo synthesis in a novel way, and this creates potential for future use in facial attribute editing, verification, and virtual media application. Moreover, the hybrid strategy also creates potential for future investigation of GAN combination with other region-based approaches so that the generative as well as the discriminative capability can be harnessed at the same time. At last, the outcome enables image synthesis in every application field to tradeoff between image quality and computation expense, and advance technical innovation and application.

1.7 Scopes and Limitations of the Study

1.7.1 Scope of the Study

This study focuses on building a hybrid deep learning model that integrates Mask R-CNN with GANs for high-resolution color face generation**.** The research process includes model design, implementation, training, and evaluation, carried out on the CelebA-HQ dataset, which provides high-quality facial images.

The scope of this study:

- Designing and building a hybrid model using Mask R-CNN for face region extraction and GANs for generating high-resolution face images.
- The CelebA-HQ dataset, which comprises high-resolution facial images, is used to train and test the model.
- Testing of the model qualitatively (visual evaluation) as well as quantitatively (PSNR, SSIM, FID) to validate its performance.
- Comparison of proposed model performance with state-of-the-art existing deep learning methods for face generation

1.7.2 Limitations of the Study

Although the proposed hybrid deep learning model is promising, it also has several limitations to mention. These constraints define the scope of this research and propose directions for future improvements:

- The study is confined to synthetic face generation, and it does not extend to real-time applications such as video synthesis or deep fake detection.
- The model was trained on CelebA-HQ, which mostly consists of celebrity faces and might not generalize well to other datasets.
- Computation limits can influence training time and hyper parameter tuning.

1.8 Contributions of the Study

The present work reports some of the most compelling results obtained from design and experimentation of the suggested hybrid deep learning architecture. The main contributions and results are summarized as follows:

- The study introduced a novel hybrid architecture that incorporates Mask R-CNN with GANs for enhancing structural coherence and naturalness of color face synthesis.
- The generator was also supplemented by channel attention and self-attention mechanisms in a manner so that the model can learn long-distance spatial relations as well as give more importance to relevant feature channels in a way that global as well as local details are preserved.
- A multi-scale PatchGAN discriminator was used, which ensured realism at different resolutions by maintaining fine detail and global structure.

- Training was guided with a combination of pixel-wise, perceptual, feature-matching, and SSIM losses to generate dramatically improved visual quality and structure consistency of outputs.

1.9 Organizations of the Study

This study consists of seven chapters, which are stated below.

Chapter One: It states an overview of the study's requirement, the issues, and the problem that motivated us to carry out this research.

Chapter Two: It discusses literature reviews, machine learning and deep learning algorithms, image-to-image translation, image generation, face synthesis, and similar past works carried out on face image generation tasks.

Chapter Three: It describes research techniques like dataset collection, preprocessing of data, development tools, baseline activity, and metrics for evaluation.

Chapter Four: It provides a precise description of the architecture of the proposed model with the generator network design and discriminator networks, model learning functions, and pseudocode describing the training procedure.

Chapter Five: It describes using the model, demonstrating a code snippet of the proposed solution, listing the hyper parameters employed, and describing the experiment class. Chapter Six: It describes the results of the proposed method and compares them to previous studies. Chapter Seven: It gives a summary of the research findings and future study recommendations.

CHAPTER TWO

2. LITERATURE REVIEW

2.1 Overview of Literature Review

Computer vision is a discipline of computer science that seeks to produce knowledge in the form of semantic information from images with the overall objective of understanding a scene (Rosenfeld, 1988). Computer vision enables computers to provide an interpretation of image data and make decisions based on them, just like the eyes and brain of a human perceive and understand the world. Computer vision encompasses methods for image acquisition, processing, analysis, and interpretation of digital images, along with high-dimensional feature extraction. With the use of artificial intelligence, machine learning, and image processing technologies, computer vision can allow systems to recognize patterns, object detection, and even interpret behavior within videos or images.

Machine learning (ML) was defined as the art of causing machines to learn and improve by themselves without special programming (Samuel, 1959). As a substitute for hardcoded rules usage, ML algorithms employ statistical techniques to enhance the ability at a task with experience. The field has developed with the advent of big data and high-performance computing that has opened up new avenues for modeling and analyzing complex, data-intensive processes (Wang, Sindagi, and Patel, 2018). Depending on the learning process and the type of data, ML techniques can be divided into three main categories: supervised learning, unsupervised learning, and reinforcement learning. Supervised learning is associated with labeled input data with known output. Unsupervised learning finds patterns or clusters within unlabeled data. Reinforcement learning is where an agent learns about an environment and determines the best actions through trial and error in the form of rewards or punishments. ML is the basis of computer vision, though image and video data are very high-dimensional and complicated, and are found challenging to standard ML techniques. Choosing helpful features will be challenging; hence, the deep learning subfield emerged.

Deep learning revolutionized computer vision, particularly with Convolutional Neural Networks (CNNs), which have gained phenomenal progress. Although initially proposed by LeCun et al. in 1989 for handwritten digit classification (LeCun et al. 1989), CNNs have emerged as the preferred

architecture due to their use in a wide range of applications, ranging from computer vision to natural language processing (Alzubaidi et al., 2021). The recent advance has also brought about incredible generative models based on CNNs that have the ability of learning complex patterns and producing new data that is barely different from the initial set of data. Among all these, the GAN is particularly famous. A generator and a discriminator are pitted against each other to reach a Nash equilibrium in order to produce remarkably realistic generated data. Goodfellow et al, introduced this game-theoretic approach for the first time in 2014.

2.2 Overview of Face Image Generation

Face synthesis is the process of creating lifelike images of human faces using deep learning techniques, particularly GANs and their variants. This task can be split into two parts: conditional generation, which allows the user to manipulate different attributes like pose, age, expression, or style, and unconditional generation, which creates completely new identities. High-fidelity, controllable, and identity-preserving synthesis across various modalities, such as RGB, sketches, and near-infrared images, has been made possible by recent developments like StyleGAN and geometry-guided GANs. For privacy concerns, face synthesis is used extensively in identity creation, entertainment, and data augmentation. The preservation of identities, disentangled control, dataset bias, and evaluation metrics are additional issues that face synthesis presents. Furthermore, Mo, Lu, and Liu (2022) have also shown how varying the spaces of colors can have significant effects on detecting and recognizing synthetic faces, bridging synthesis research with the growing need for strong synthetic face detection systems.

2.3 Face Image Generation Techniques

In recent years, facial color image generation has advanced significantly, mostly as a result of improvements in deep learning algorithms. Variational Auto encoders (VAEs), GANs, and hybrid models which combine several techniques for increased accuracy and realism.

2.4 Generative Adversarial Networks

A GAN is a computer learning model based on earlier research by (Goodfellow et al. 2014) on noise contrastive estimation and loss functions in the existing models (Grnarova et al. 2019). The practical application of GANs began to occur around 2017, particularly in the generation of high-quality human face images, which was useful in image enhancement and illustration quality.

Although the concept of adversarial networks was originally proposed in an Olli Niemitalo blog in 2010, the approach can also be termed as Conditional GAN.

Generative models are also a type of deep learning method. Deep learning is an approach that utilizes neural networks with an unspecified number of layers. It is considered a subdomain of machine learning. The techniques are widely used in applications such as image recognition, voice synthesis, and data mining, where hierarchical models are constructed to represent the probability distributions of data. In wireless communication networks, deep learning in the form of conditional GANs with DNNs can perform encoding, decoding, modulation, and demodulation. Directing DNN operations requires a precise, real-time evaluation of the channel transfer status (Ye et al., 2018).

GANs, which were formulated for the first time in the vicinity of 2013 by researchers attempting to replicate animal behavior, are a new deep learning model. They function through the employment of two neural networks: a generator, which produces images, and a discriminator, which checks for their validity. For example, in identifying real and fake images, the generator produces fake images that are similar to real images, while the discriminator attempts to differentiate between real and generated images (Hsu, Zhuang, and Lee 2020). The networks tend to use CNN architectures. Training continues until the discriminator can no longer consistently distinguish real and generated images as real or generated, i.e., the generator has learned to generate very realistic descriptions. Generated images may appear realistic to audiences and occasionally exhibit characteristics of real images (Marra et al. 2018). GANs are applicable in supervised learning and unsupervised learning, and also in reinforcement learning, where the generator produces candidate images and the discriminator verifies them.

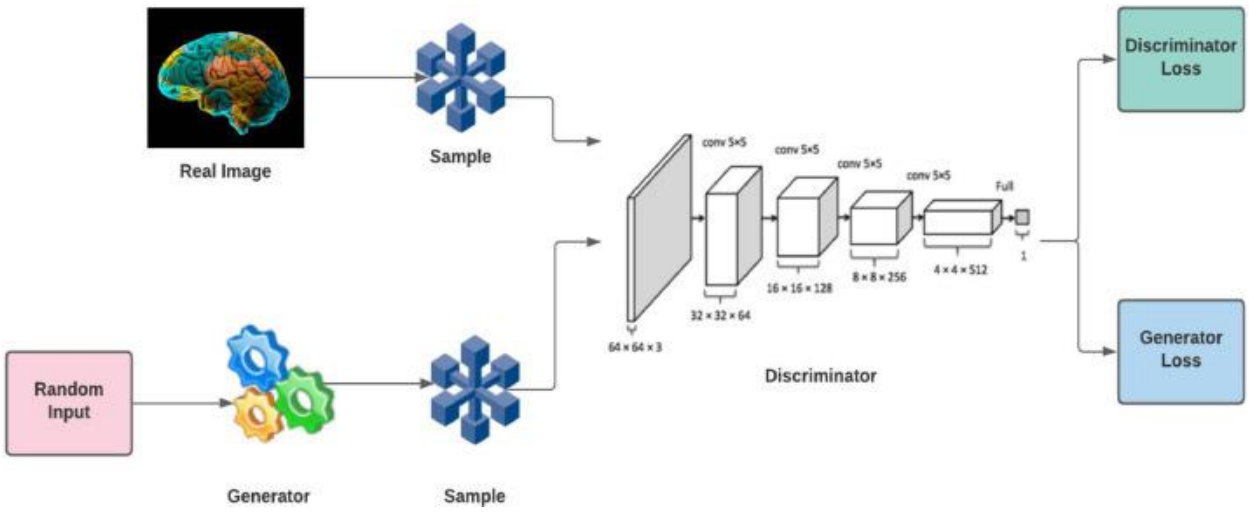


Figure 2. 1 Block diagram of the Generative Adversarial Network

GANs were originally introduced (Goodfellow et al. 2014) based on game theory, where two neural networks, the discriminator and the generator, are involved. They play against each other to reach a Nash equilibrium. The generator tries to produce facial images as realistic as possible in order to fool the discriminator into classifying them as real. Meanwhile, the discriminator evaluates the images and forwards the feedback to the generator, which helps the generator improve its ability to produce realistic fake data that can fool the discriminator (Figure 2.1). The two networks engage in a min-max game, defined by an exact value function (Eq. 2.1).

$$\min_G \max_D \left[\mathbb{E}_{x\sim p_r}[\log(D(x))] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]\right] \qquad \text{(Eq. 2.1)}$$

Where G and D represent the generator and discriminator models, respectively, the generator attempts to approximate the probability distribution of the samples given an input equation, producing data that closely resembles real samples.

2.5 Generative Adversarial Networks Architectures

2.5.1 Deep Convolutional GAN (DCGAN)

DCGAN is the most popular and applied type of unsupervised learning method. In DCGAN, CNNs are used in place of the Multilayer Perceptron (MLP) with additional constraints. The effectiveness

of Convolutional Neural Networks in both unsupervised and supervised learning has been enhanced with the introduction of DCGAN. Figure 2.2 depicts the Generator structure of DCGAN. The discriminator is the inverse of the Generator, i.e., it takes input image data and produces two numbers. The forward process is used on the Discriminator side, which consists of the Conv Transpose or Deconv processes at each (S and Durgadevi 2021).

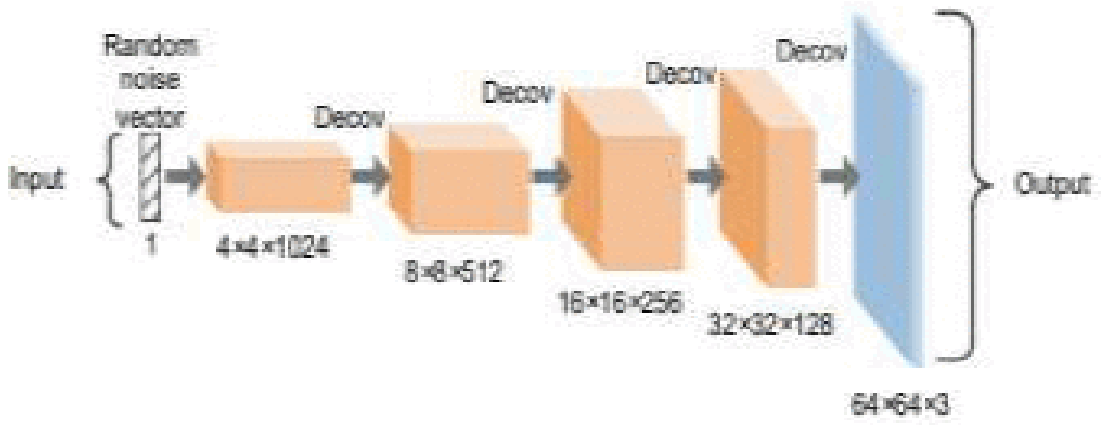


Figure 2.2: Architecture of the generator in DCGAN(S and Durgadevi 2021)

2.5.2 Fully Connected GAN (FCGAN)

Goodfellow (Goodfellow et al. 2014) proposed his actual GAN, which uses FCGAN for the Discriminator as well as the Generator. Normally, the FCGAN method is used most of the time on some simple image datasets like the Toronto Face Dataset, CIFAR-10, and MNIST. Good generalization is not provided for more complex images in Fully Connected Neural Networks(S and Durgadevi 2021).

2.5.3 Conditional GAN (CGAN)

Conditional GAN is a type of supervised learning GAN method, in which each Discriminator and the Generator are conditioned to some extra auxiliary data 'C'. In the Generator side, the input noise 'Z', and auxiliary data 'C' are mixed and trained together as shown in Figure 2.3. The framework is used to check the flexibility of how the hidden representation 'C' is composed. This information consists of class labels or data from other modalities. In general, the network structure is a Multilayer Perceptron (MLP) for CGAN(S and Durgadevi 2021).

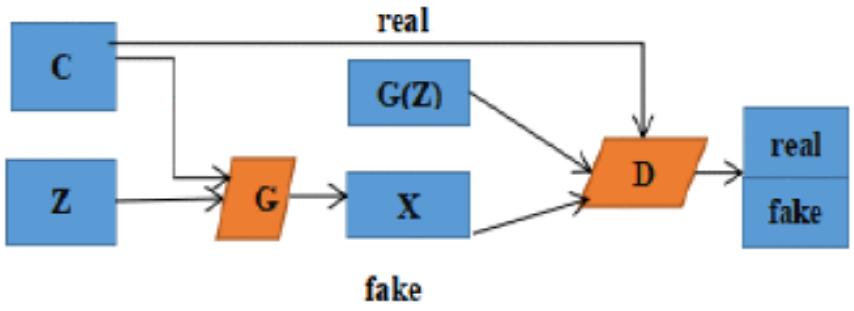


Figure 2.3: Conditional GAN (CGAN)(S and Durgadevi 2021)

2.5.4 CycleGAN

The objective of image-to-image translation is to learn a mapping between an input image and a corresponding output image using a training set of aligned image pairs. However, in many cases, such paired training data is not available. CycleGAN (Heusel et al. 2017) proposed a method for learning to convert an image from a source domain X to a target domain Y without the need for paired examples. It is a type of generative adversarial network for unpaired image-to-image translation. For two domains X and Y, CycleGAN (Heusel et al. 2017) learns a mapping G: X→Y and F: Y→X. The innovation is in attempting to ensure that these mappings are inverses of each other and that both mappings function as bijections. This is accomplished through a cycle consistency loss that promotes F(G(x)) ~ x and G(F(y)) ~ y. Combining this loss with the adversarial losses on X and Y yields the full objective for unpaired image-to-image translation.

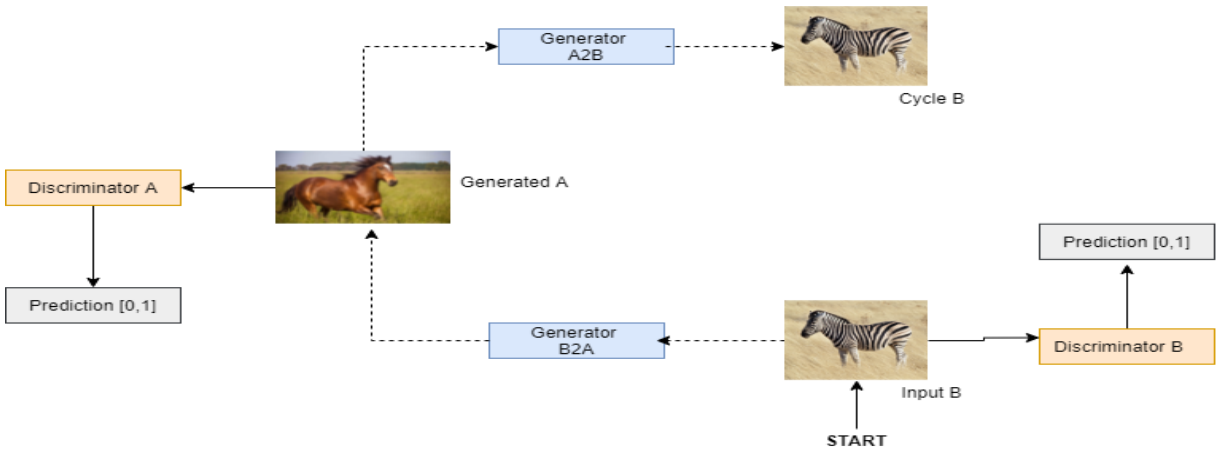


Figure 2.4: Cycle GAN (Zhu et al., 2017)

2.6 Mask Region-based Convolutional Neural Networks (Mask R-CNN)

Region-based approaches, specifically Mask R-CNN, are typically applied in segmentation tasks, allowing the model to efficiently learn spatial correlations between face regions. Mask R-CNN was discovered to be effective in precisely localizing and segmenting the region of interest (Heusel et al. 2017), and thus is applicable in scenarios where spatial precision as well as object localization are paramount, e.g., facial attribute detection and medical imaging (Krizhevsky, Sutskever, and Hinton 2012).

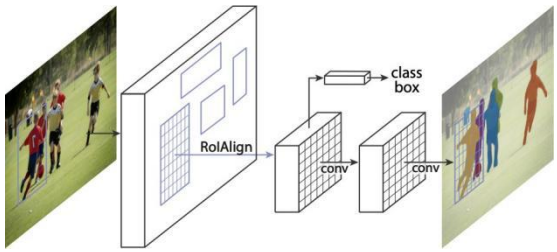


Figure 2.5: The Mask R-CNN framework for instance segmentation (He et al., 2017).

Mask R-CNN and GANs combination holds tremendous potential for the enhancement of feature localization. The combination of Mask R-CNN and GANs, according to the latest research as put forward by Chen et al., not only improves the quality of the produced image, but also accuracy in feature segmentation (Shu et al. 2020). Similarly, GAN hybrid models that integrate GANs with region-based models have become more popular for their ability to improve the realism and fidelity of output images. Zhang et al. (Daras et al. 2020), for example, emphasized the ability of attention in GANs to drastically enhance the attention of the model towards the most important facial regions. Additionally, Mirza and Osindero's (2014) work on conditional GANs was found to condition that could restrict image generation to a region, and it is very well-suited for localized face enhancement if combined with Mask R-CNN.

From a measuring perspective, generative models are generally measured in terms of scores comprising the FID, which approximates distributional closeness between real and generated images within the latent Inception network space (Heusel et al. 2017). FID is presently regarded as the de facto metric, with studies by Salimans et al. (2016) showing low FID scores to be strongly correlated with higher perceptual quality. Besides other assessment factors, FID enables a similar comparison of generative models and facilitates improvement on an ongoing basis in image generation.

According to the improvement gained, the paper proposes a hybrid Mask R-CNN and GAN model designed particularly to generate high-quality, realistic-color face images. By harnessing the precise region-level segmentation of Mask R-CNN with the generation ability of GANs, the approach is bound to outperform single-GAN models.

The new hybrid method provides a fresh approach to synthetic face generation that can find applications in augmented reality, face recognition, and other areas using high-fidelity facial images and is likely to capture the global facial shapes as well as local facial structures.

2.7 Translation of Image to Image (I2I)

For instance, when we have a selfie image and we wish to convert it into a more creative form, such as a cartoon drawing, such research is referred to as image-to-image translation. I2I translation attempts to transfer images from one class to another without losing the content representations (Pang et al., 2022). Image-to-image translation processes can be implemented through supervised training or unsupervised training settings. The 19 supervised I2I translations require a set of pairs of sample datasets to project source images into target images. Constructing a dataset for all the matched samples can improve translation performance with supervised I2I translation, although it is difficult and expensive. Unsupervised I2I translation overcomes the problem of translating a given image from one source to another using paired training samples (Shukla et al., 2019). I2I translation is gaining popularity today since it can be applied in various computer vision applications, such as image colorization, face generation, and face recognition.

2.8 Face Recognition

Human facial appearance is a window to a person's internal state. It is able to provide a lot of information, like the person's mood, what a person is planning to do with others, and how concentrated an individual is. Other than these emotional cues, facial features are also significant when it comes to recognizing a person. Facial recognition is a field within computer vision that is tasked with automatically identifying people based on digital images or video. This is not only a matter of face presence detection but also the extraction and analysis of its characteristic features. In order for proper recognition, the system must be invariant to changes in factors like illumination, facial expressions, pose, aging, and even minor image transformations. However, image translation, rotation, scaling, and pose are some of the hard problems in face recognition (Parmar & Mehta, 2014). It is now one of the most popular biometric techniques used for identity

authentication, being utilized in areas such as military, finance, public security, and even in everyday life.

2.9 U-Net

The U-Net model, initially introduced by Ronneberger et al. (2015), is a powerful convolutional neural network used mainly for semantic segmentation. With its encoder-decoder structure supported by skip connections, it has shown outstanding performance in a very general class of image-to-image translation and reconstruction problems, thereby making it extremely well-suited to produce high-fidelity images. By integrating hybrid attention mechanisms in the U-Net, it is even more powerful in concentrating on key features at multiple scales. Such integration is particularly helpful for sophisticated image transformations, i.e., transforming slanted iris images to real frontal ones, where precise rendering of fine textures and structural details is most important for achieving high-quality outputs.

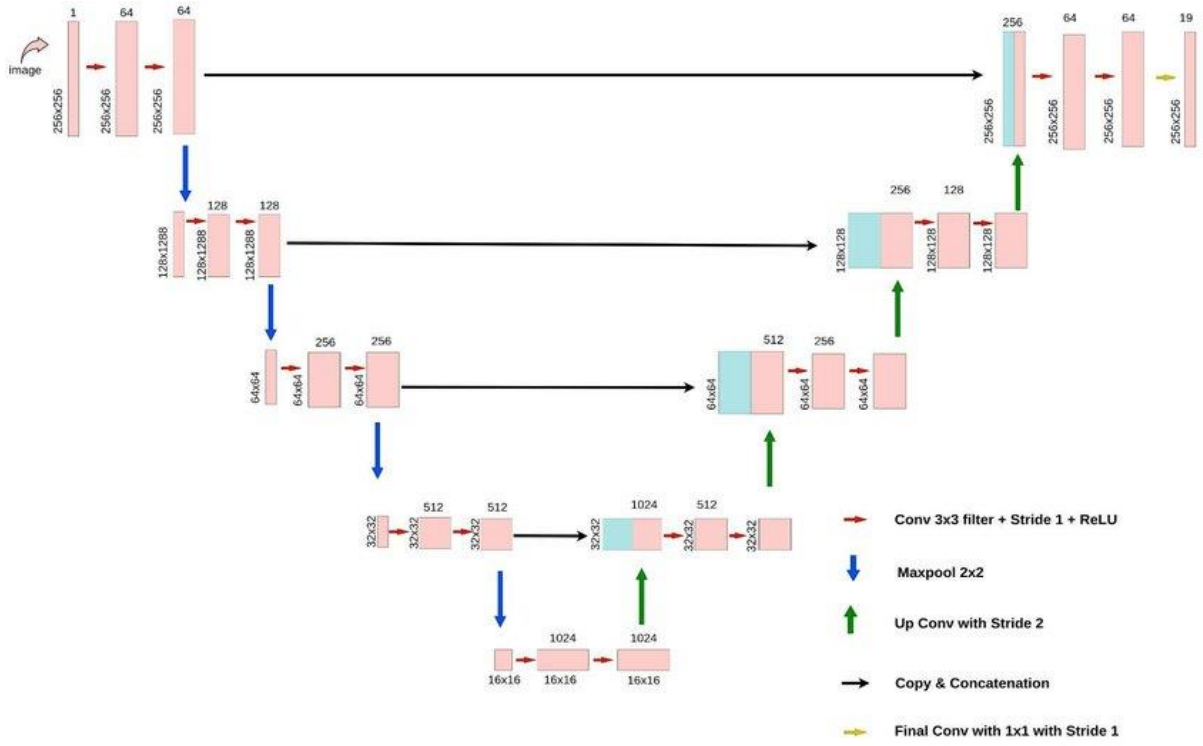


Figure 1.6 U-Net Architecture

2.10 Related Work

Chang et al. (2024) introduce an innovative GAN that produces colored portraits from the input sketch. The process utilizes GPU computing for efficient processing and training. Comparative

multi generation comparison is the context in which authors explain the suggested GAN model. They suggest creating multi-skin tone portraits through the preparation of a face dataset with labels according to skin color and ethnicity. This is done in an attempt to restrict the shortcomings of normal networks that are trained on a single skin tone. Kaushik and Singh (Kaushik and Singh 2022) propose Deep DCGANs in generating images of human faces. Backpropagation and competitive training and a generator (G) are employed in generating imposter images, using a generator (D) which will be capable of distinguishing between real and imposter images. The model generates realistic human faces from raw data and noise. The model is effective with the SSIM and PSNR measures.

Guo et al. (2019) introduce the AEGAN generating high-resolution images of higher perceptual quality. Traditional GAN-based models remove high-resolution detailed complex information. Thu s, they can generate foggy or incomplete structures. AEGAN solves the issue by employing an auto encoder to preserve the image structure and a denoising network to produce more detailed details and photorealism. The experimental result verified that AEGAN could learn to produce 512×512-pixel images with better-structured sharpness and more contents than baseline approaches. It was tested on different datasets including Oxford-102 Flowers, Caltech-UCSD Birds, CelebA-HQ, LSUN, and ImageNet and produced better image realism and quality.

Karras, Laine, and Aila (2019) presented StyleGAN, a high-res face generation model with controllable attributes like pose, expression, and lighting. Using a style vector within generator layer feature map synthesis, StyleGAN generates extremely realistic human faces and it is an improved generative model. Its weakness in generating portraits with a single skin tone is that it is not highly variable. Future studies will be dedicated to increased ability to build portraits in various skin tones and races with increased inclusivity.

Miranda-Hernández, Castelán, and Torres-Méndez (2012) proposed a face color synthesis method through the application of Partial Least Squares (PLS) regression hybrid and luminance-α-β color space conversion. The existing synthesis methods suffer from loss of uniformity of colors with varying lighting conditions, thus unrealistic. PLS approximates the congruence of face colors and luminance variation by their method. The distortion minimization is also utilized using luminance-

α-β space. The method acquires certain importance in computer vision and face recognition. Realistic and authentic skin tone description is of utmost concern both for realism as well as usability.

Wang, Sindagi, and Patel (2018) introduced a multi-adversarial network for generating photorealistic photo-sketch synthesis. This article attempts to break the limitation of traditional methods. Traditional methods lack necessary structure information and possess accuracy problems as far as precision is concerned. They use more than one generator and discriminator in such a manner that they are able to produce more realistic and accurate sketches without losing facial features. This attempt is central to applications of face recognition, entertainment, and security where high-quality sketching is needed.

Sun et al. (2022) suggested a face-to-sketch translation using a generative adversarial fusion approach. It focuses on structural accuracy instead of aesthetic appeal. Other techniques cannot provide accurate data and authentic facial edges and hence yield sketches with too little plausible information. Their method utilizes a chain of adversarial networks to make the generation process even longer and generates aesthetically pleasing and structurally accurate sketches. This is utilized in face recognition, computer graphics, and security all which need high quality sketch conversion.

Table 2. 1: Related Work in Color Face Generation

Related Papers	***Datasets***	***Methodology***	***Contributions***	***Limitations/Gap***	***Gap Addressed in Current Study***
Wang et al. (2018)	CUHK Face Sketch, CUFSF	Multi-adversarial network for photo-to-sketch generation.	Enhances realism and structural accuracy in facial sketches by employing multiple adversarial networks, resulting in more detailed and realistic outcomes.	Applicable only to photo-to-sketch conversions; may not generalize well to other imagetypes.	Current study focuses on color face generation rather than sketch, addressing generalization for real face synthesis
Chang et al. (2024)	CelebA-HQ	Improved GAN with GPU-based training and image generation	Proposed an improved GAN model to enhance the quality and diversity of synthesized color face images. Addresses limitations of standard GANs in capturing realistic color and texture for various skin tones and facial features.	Primarily focused on color face generation; adaptability to a more complex image domain is uncertain	Uses Mask R-CNN for semantic segmentation to improve structural and color accuracy
Wang et al. (2018) Pix2PixHD	Cityscapes	Multi-scale GAN + feature matching + perceptual loss	High-resolution and photorealistic synthesis	Focused on scenes, not face-specific; lacks semantic face segmentation	Applies instance-level face segmentation for accurate facial detail synthesis

Miranda-Hernández et al. (2012)	Custom datasets	Partial Least Squares regression integrated with luminance color trans form.	Improves facial color synthesis and ensures more even skin tone representation under changing lighting. Enhances facial recognition and color accuracy.	May not fully address all lighting variation challenges in complex or dynamicenvironments.	Current study uses GAN with attention mechanisms to improve realism under variable conditions
Sun et al. (2024)	AffectNet	Multi-scale Mixed Attention GAN with residual connections and mixed attention module	Enhances synthesis of facial expression, image quality, and realism.	Synthesized faces often show blurring or overlapping features, resulting in unnatural expressions	Addresses fine-grained facial feature preservation using Mask R-CNN guidance
Karras et al. (2019)	FFHQ, CelebA-HQ	StyleGAN with a "style" vector to control features like pose, expression, and lighting.	Enables fine-grained manipulation of image features and generates photorealistic human faces at high resolution. Expands the Capability of generative models in portrait synthesis.	Struggles to produce ethnically diverse and multi-toned facial images. Focused only on makeup style transfer; lacks conditioning on structural face masks.	Introduces Mask R-CNN semantic masks to guide GAN and improve structural fidelity
He et al. (2017)	MS COCO, Face detection datasets	Instance segmentation via ROIAlign + FCN	High-accuracy object and face region extraction	Not generative; needs integration with GANs for synthesis	Current study integrates Mask R-CNN with GAN for color face generation

Zhang et al. (2019) SPADE (Semantic Image Synthesis)	COCO, ADE20K	Spatially-adaptive normalization with semantic maps	Semantic-to-image synthesis with fine control	Needs pixel-wise masks; no instance-level face handling	Current study uses instance-level Mask R-CNN masks to improve semantic precision

Table 2.1 lists the most applicable prior studies in color face generation with the datasets, approaches, contributions, and limitations of each paper. The preliminary studies began with sketch-to-photo translation, followed by more advanced models based on GANs that improved the realism and diversity of the faces. Other studies utilized semantic-based methods, region-aware learning, or attention mechanisms to improve facial image synthesis. Although such approaches have made significant contributions to the research, they are still hampered by low generalizability, lack of structural consistency, inadequate handling of various skin colors, and insufficiency of proper preservation of fine local features such as eyes, mouth, and hair. Segmentation-based methods provide accurate region extraction but do not have the inherent generativity, which indicates the absence of structural guidance and image synthesis cohesion in existing approaches. To complement such limitations, the present work presents a hybrid deep learning model integrating Mask R-CNN and GANs for generating color faces. With the help of precise instance-level segmentations as structural priors, the model at hand enhances boundary precision and enforces structural coherence along the generation process. The GAN component is subsequently encouraged to produce photorealistic faces with more texture, color accuracy, and high-resolution local detail preservation. The method also tries to promote diversity and fairness in generated faces by pushing improved representation within facial attributes and skin tones. According to this, the hybrid Mask R-CNN + GAN model aims directly to address the limitations of current approaches towards structure-aware, high-fidelity, and multi-color facial synthesis.

CHAPTER THREE

3. RESEARCH METHODOLOGY

3.1 Overview of the Chapter

This chapter outlines the methodology adopted to carry out this research. It describes an overview of the steps carried out to develop and test a color face generation model. It discusses datasets and the preprocessing techniques performed. Finally, it mentions evaluation metrics, software libraries, and hardware utilized for developing and training the model. The method adopted in conducting this thesis includes defining the area of research, conducting a review of the literature from other researchers, reviewing the gaps in past research, developing research questions, choosing the right tools and techniques, preparing and gathering datasets, and lastly designing and implementing the proposed solution. Figure 3.1 illustrates the general research design process overview from the identification of the research area to model development, model evaluation, and report writing.

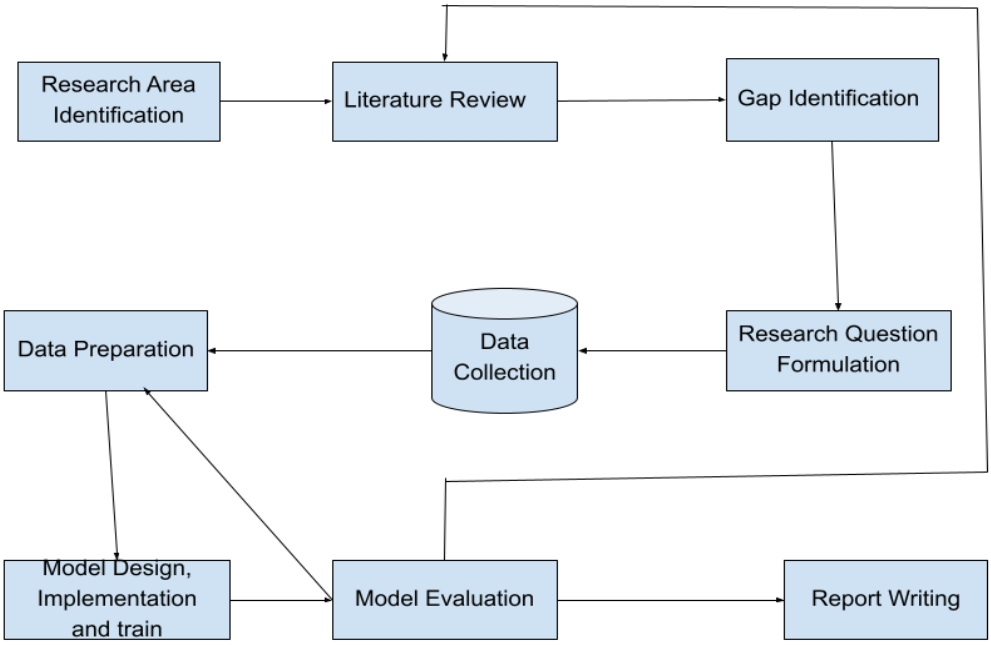


Figure 3. 1: Overall Research Process

3.2 Description of Study Area

This work presents a thorough exploration of the state-of-the-art deep learning techniques for the best performance ever for facial image synthesis, intending to combine Mask R-CNN with GAN. The work tries to synthesize high-resolution, color-correct face images with fine-grained facial detail in primary facial regions, especially under partial occlusion by some level of masking. Mask R-CNN also carries out face localization, along with facial sharpening of significant facial features such as the nose, mouth, and eyes. Some of the application areas of the work include occlusion-based face recognition, media, virtual avatar creation, augmented reality, and even medical imaging, where realistic face generation can be beneficial.

3.3 Datasets

The research focused on the challenge of facial generation using state-of-the-art deep learning techniques. Deep learning models significantly rely on being provided with data to learn perfectly. Data was therefore at the heart of our research work. Mask RCNN and CelebA-HQ datasets were utilized for training and testing in face generation using GAN. This data set was picked specifically for being of the very best quality and having a very broad variation of facial traits.

CelebA-HQ is a high-resolution extension of the original CelebA (Celebrities Attributes) dataset (Liu et al. 2015) with over 200,000 celebrity faces and labeled facial attributes. In contrast to the lower resolution (178×218 pixels) image data in the original, CelebA-HQ offers 30,000 high-quality images resized to a resolution of 1024×1024 (Karras, Laine, and Aila 2019), which experienced enhanced visual quality with the elimination of duplicates, artifacts, and low-quality aligned faces. All images are labeled with the same 40 binary attributes, such as gender, smile, eyeglasses, and facial hair (Zhang et al., 2018), which makes it an excellent case of facial attribute classification and applications of generative models. The dataset is widely utilized in deep learning and computer vision communities to train GANs like StyleGAN (Karras, Laine, and Aila 2019), besides other applications like face editing, super-resolution, and landmark localization. Even though the raw CelebA dataset can be accessed on websites such as Kaggle and the MM Lab website, the variation between CelebA-HQ and the raw images most likely calls for processing or even particular scripts (Hsu, Zhuang, and Lee 2020). Due to its increased quality and reliability, it has been used as a normative reference to consider research regarding face image synthesis and analysis (Wang et al., 2021).

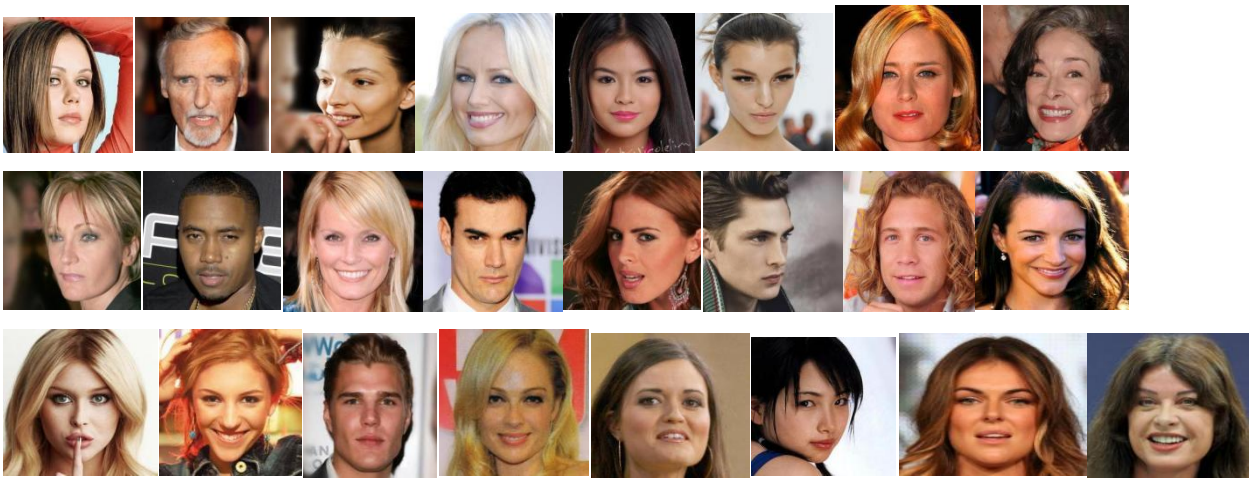


Figure 3.2: Dataset Samples

The CelebA-HQ dataset was chosen for its superior visual appearance, structured alignment, and full facial features, which are crucial to the training stability of GANs and realism of generated outputs. However, it comprises an inherent celebrity bias, which limits the diversity of ethnicity, age, and surrounding conditions. In spite of the higher ethnic and photo diversity options such as FFHQ and DIV2K, they were not employed in this research due to their weaker labeling consistency and less precise face alignment, considerations that are determinant for Mask R-CNN segmentation precision.

3.4 Pre-processing Techniques

Preprocessing is essential for good-quality color face synthesis because its conditions images to suit the model's requirements. Preeminent amongst these steps is resizing and normalizing the image, which provides consistency and standardization throughout the dataset.

3.4.1 Image resizing

Because the dataset can have images of varying sizes, it's necessary to resize all the images in one format. Resizing images is an important deep learning preprocessing that enables input data to be fitted into the given network size. In particular, images should be resized to 256x256 pixels. Such standardization is easy to process in batches, so that input to a model can be convenient. But inaccurate resizing can result in loss of information as well as deformation of the image, and therefore should be accomplished with caution.

3.4.2 Normalization

Normalization is a simple preprocessing that normalizes the pixel intensity of images typically to a standard range such as [0, 1]. Normalization standardizes all the images with the same dynamic range and prevents the generative model from becoming biased by the image-to-image differences in the pixel intensity. This is an important feature of GANs since it enables the network to learn meaningful features without distraction from variation differences in pixel values. Normalization also prevents outliers or high pixel value instances that could destabilize training and lower image quality otherwise. Overall, application of normalization is found to improve model convergence, training stability, and assist in generating face images of higher color quality and realism.

3.4.3 Data Augmentation

Data augmentation techniques are strong machine learning techniques that are used to artificially increase the size and diversity of a training set. By converting current data into other forms of data, they add diversity, allowing the model to train on a far greater diversity of situations than would otherwise be seen. Introduction of variability is useful since it renders and makes the model robust and adaptable when operating in varying conditions and hence protects it from overfitting. In this research, we used a number of augmentation techniques to improve our dataset. These included horizontal flip to produce mirror images and thereby train the model's orientation invariance; random rotation to alter the view of images and mimic other views; and brightness, contrast, and saturation changes to compensate for lighting differences. The ultimate aim of this enrichment process was to generalize the model, and it then leads to improved performance on new data as well as in practical application (Krizhevsky, Sutskever, and Hinton 2012).

These preprocessing steps were not trivial but were specifically intended to stabilize GAN training. GANs are extremely sensitive to input magnitude and distribution of pixel values; different image sizes or non-normalized pixel ranges will result in generator and discriminator imbalance, either in the form of mode collapse or unstable convergence. Rescaling to 256×256 is a compromise between computational feasibility and sufficient feature resolution for facial details. Normalization makes gradient flow stable during adversarial optimization, and augmentation leads to better generalization towards various facial orientations and light conditions, having a direct impact on the realism and variety of synthesized color faces.

3.5 Data Partitioning

The dataset for the study is 10,000 face images. In order to train and test the model efficiently, data were split into training, validation, and test sets. The training set had 75% of the total data (7000 images) and was used for the tuning of the hyper parameters of the model. The validation set had 15% of data (1500 images) and was used to tune hyper parameters and prevent overfitting. The remaining test set, i.e., 15% of the data (1500 images), is kept aside as a hold-out set for end model performance evaluation. Such partitioning allows strict testing of the generalization ability of the model on unseen data.

3.6 Hardware and Software Requirements

A range of software and hardware tools was utilized in the design and implementation of this study.

3.6.1 Software Requirements

Software tools aid in numerous phases of the research process, ranging from model building and data preprocessing to visualization and deployment. The software used in this study is summarized in Table 3.1

Table 3. 1: Software Tools

Software	***Description***
Anaconda	An open-source package that bundles Python and R with a multitude of scientific computing, machine learning, and data science libraries. It simplifies package deployment and administration by bundling several open-source packages into a single installation.
Jupyter Notebook	An interactive web-based notebook interface that enables users to create and share documents that contain executable code, equations, visualizations, and narrative text.
PyTorch	A free machine learning platform commonly used for developing and training neural networks for various machine learning applications.
EdrawMax	Presentation, analysis, and design software are used here to create system diagrams and illustrations that are relevant to the work proposed.
Google Drive	A cloud data storage service where the datasets have been kept and shared to make preprocessing and training possible on Google Colab.

Google Colab	No-install, cloud-based Jupyter Notebook environment available for free use, granting access to GPUs and other computational resources for deep learning operations.
CUDA	NVIDIA's parallel computing platform and programming model enable effective use of GPUs for deep neural network computations.

3.6.2 Hardware Tools

The hardware components used in this study are summarized in the table below:

Table 3. 2: Hardware Tools

Tools	***Description***
GPU	NVIDIA GeForce RTX GPU for accelerating the training of deep learning models.
RAM	Employed to enhance computational speed and performance by enabling seamless dataset loading and efficient patch-based image processing.
Hard Disk (HDD)	Used to store facial image datasets, trained models, experimental results, and research findings

3.7 Baseline Work

The baseline model from this work is employed for comparison purposes in relation to architecture to check how the improvements brought through the hybrid framework here stand in comparison with it. It is a typical Generative Adversarial Network architecture that involves a U-Net generator and single-scale PatchGAN discriminator, borrowed from Isola et al. (Isola et al. 2017). The U-Net architecture is commonly used in image-to-image translation tasks since it involves a symmetric encoder-decoder network and skip connections that allow the retention of low-level spatial information that is important for fine-grained reconstruction of images (Ronneberger, Fischer, and Brox 2015). The discriminator is a single-scale discriminator that tries to identify the synthetic or real color face images using local patches only. This base model accepts a sketch or grayscale input and attempts to synthesize a natural-color facial image. It possesses no advanced mechanisms for capturing global dependencies or injecting semantic structure, and so is apt to

create artifacts, incorrect colorization, or structural deformities in the output. Despite these limitations, the baseline is required for comparison of the improvements through the addition of an attention mechanism (Zhang et al. 2019) and semantic mask guidance with Mask R-CNN (He et al. 2017).

Baseline U-Net generator and PatchGAN discriminator were used as per their being a well-defined, explainable, and comparatively state-of-the-art image-to-image translation model. It offered a good enough test bed for demonstrating quantifiable advancement in terms of hybridization. More sophisticated models, such as StyleGAN or Pix2PixHD, offer superior synthesis quality but also include very expert constituents that would water down the specific strengths of Mask R-CNN integration. As such, the chosen baseline provides a platform for an equitable comparison and better performance gain, which favors the suggested hybrid framework.

3.8 Feature Extraction Network

The feature extraction network play a vital role in improving the generator's performance by providing rich semantic and structural information about the input face image. In our work, a pre-trained Mask R-CNN is employed as the semantic feature extractor to generate facial masks that guide the generator in focusing on important facial regions such as the eyes, nose, mouth, and jawline(He et al., 2017). For the generator to learn localized and context-aware representations during training, these masks are concatenated with the sketch input to create a semantically enriched input.

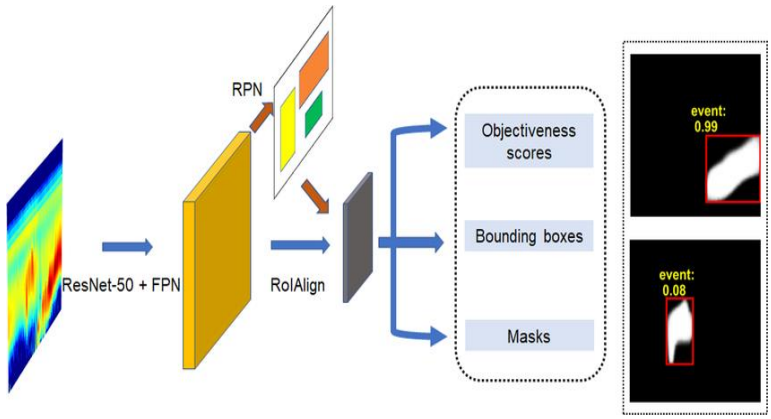


Figure 3. 3: Mask R-CNN Architecture

The choice of utilizing Mask R-CNN as the core model for segmenting images followed the fact that it provides instance-level segmentations with state-of-the-art pixel-level accuracies, which play a critical role in maintaining detailed facial features, including the skin contours, eyes, and lips. Although other models, U-Net (Ronneberger, Fischer, and Brox 2015) and Deep Lab (Bharati and Pramanik 2020), perform best when it comes to carrying out the semantic segmentation operations, they prove incapable when separating overlapping or close-set facial features, which ends up producing blurred or misplaced edges. Similarly, compact models like YOLACT (Bolya et al. 2019) yield fast times when performing the inference operations, but at the sacrifice of the accuracy level, a factor that makes them less effective when creating high-fidelity facial images. Earlier research works highlighted the fact that including Mask R-CNN as part of the GAN architectures increases the level of feature localizations and generates images with increased realness (Chang et al. 2024). Consequently, the Mask R-CNN model was utilized during the current research as a means of generating strong, semantically accurate masks, facilitating the generator when creating structurally well-defined and realistically colored facial images.

In addition to semantic segmentation, a pre-trained VGG19 network is used as a perceptual feature extractor for computing perceptual loss, which compels the generator to create images not only visually real but also semantically consistent with the ground truth (Simonyan and Zisserman 2015). The VGG-based perceptual loss compares the high-level feature maps of generated and real images between different convolutional layers to identify texture and style correspondences beyond pixel-level realism (Johnson, Alahi, and Fei-Fei 2016). Together, the networks for feature extraction assist in generating structural realism and perceptual quality in color face images synthesized by incorporating region-specific and global contextual information.

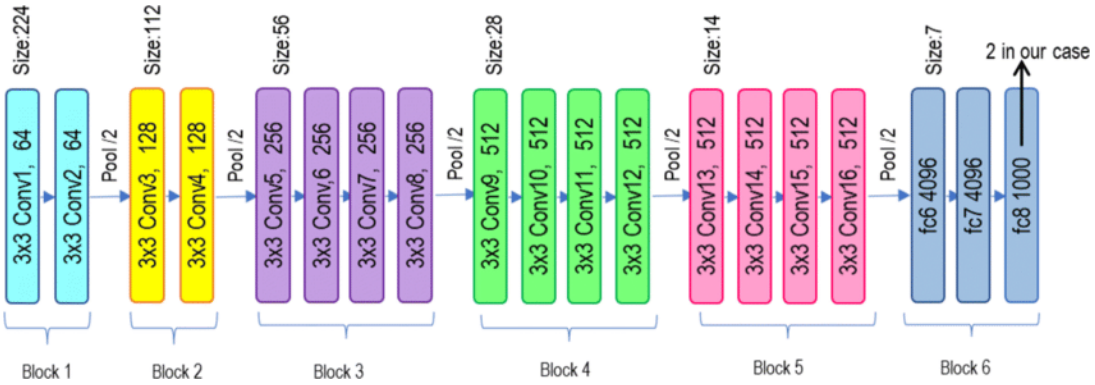


Figure 3. 4: VGG16 Architecture

3.9 Evaluation Method

In this paper, the model's performance was assessed using a combination of objective metrics like FID, SSIM, PSNR, and human perceptual evaluations, to provide a comprehensive measure of image quality.

3.9.1 Structural Similarity Index Measure (SSIM)

Structural and visual similarity to real images is assessed using SSIM. It is a metric widely used to measure the similarity between two given images and quantify image quality degradation caused by image processing. It compares the original and output images across three primary attributes: luminance, contrast, and structure.

$$\text{SSIM}(i,j) = \text{SIM}(i,j) = \frac{(2\mu_i\mu_j + C_1)(2\sigma_{ij} + C_2)}{(\mu_i^2 + \mu_j^2 + C_1)(\sigma_i^2 + \sigma_j^2 + C_2)} \qquad \text{(Eq. 3.1)}$$

Where:

- ✓ μ_i, μ_j are the mean intensities of images i and j,
- ✓ σ_i^2, σ_j^2 are the variances,
- ✓ σ_{ij} is the covariance,
- ✓ C_1 and C_2 are constants to stabilize the division.

3.9.2 Peak Signal-to-Noise Ratio (PSNR)

PSNR computes image quality in the form of quantifying noise within synthesized images. PSNR quantifies the extent to which a reconstructed image or signal may resemble the original with noise elimination and detail retention. PSNR is quantified computationally as a ratio between the maximum possible signal power and noise power that depreciates image quality. The metric is expressed in decibels (dB) so that higher values indicate improved reconstruction and improved image enhancement (Isa et al., 2015). Similar to SSIM, PSNR requires both the synthesized image and its corresponding ground truth image as input. Here in this research work, SSIM along with PSNR, are employed in the evaluation phase to take advantage of different dimensions of image quality: whereas SSIM emphasizes structural similarity, PSNR addresses pixel-level accuracy, reflecting a broader perspective about the quality of image reconstruction or synthesis.

$$\text{PSNR} = 20 \cdot \log_{10}\left(\frac{\text{MAX}_I}{\sqrt{\text{MSE}}}\right) \qquad \text{(Eq. 3.2)}$$

Where:

- ✓ MAX_I denotes the maximum possible pixel intensity value of the image (e.g., 255 for an 8-bit image),
- ✓ MSE stands for Mean Squared Error, which measures the average squared difference between the original and the synthesized (or reconstructed) image.

$$\text{MSE} = \frac{1}{M \cdot N}\sum_{i=1}^{M}\sum_{j=1}^{N}(f(i,j) - g(i,j))^2 \quad \text{(Eq. 3.3)}$$

Where:

$f(i,j)$ is the pixel value at position (i,j) in the original image, $g(i,j)$ is the pixel value at position (i,j) in the synthesized (or reconstructed) image, $M \cdot N$ is the total number of pixels in the image.

3.9.3 Fréchet Inception Distance (FID)

FID is the 2-Wasserstein distance between real image distribution P_r and generated distribution P_g. Its computations require two input sets: a collection of real images and a corresponding set of images that have been created by a GAN. FID is typically applied after the training phase in the architecture to measure how realistic and varied the generated image set is compared to the real set. It computes similarity between these two distributions to estimate the quality of generated images and is characterized by

$$\text{FID}(p_r, p_g) = \frac{\| \mu_{pr} - \mu_{pg} \|^2}{2} + \frac{1}{2}\,\text{Tr}\left(C_{pr} + C_{pg} - 2\left(C_{pr}C_{pg}\right)^{\frac{1}{2}}\right) \qquad (Eq.\ 3.4)$$

Where:

- ✓ μ_{pr}, μ_{pg}: Mean vectors of the feature representations of the real and generated images, respectively.
- ✓ C_{pr}, C_{pg}: Covariance matrices of the feature representations of the real and generated images, respectively.

- ✓ Tr(·): Trace of a matrix.
- ✓ $\left(C_{pr}C_{pg}\right)^{1/2}$: Matrix square root of the product of the two covariance matrices.

3.9.4 Human Perceptual Evaluation

It's a qualitative method used for quantifying image or signal quality. It employs human participants to judge synthesized or reconstructed images on realism, visual quality, and overall performance. As opposed to objective measures such as FID, SSIM, or PSNR, which provide quantitative values from structural or statistical dissimilarity, perceptual evaluation comprises human judgment regarding the image quality. This second assessment is crucial in use cases such as face creation, where mathematical metrics at times may not capture nuance, naturalist, and attractiveness but are included in assessing the real-world performance of the model.

A selection of SSIM, PSNR, and FID was made judiciously to correlate corresponding image quality dimensions. PSNR focuses on pixel-level reconstruction that deals with precision in preserving fine detail at low-level textural resolution. PSNR alone is insufficient because high PSNR is not always commensurate with perceptually realistic output. SSIM, on the other hand, deals with structural similarity and hence is particularly effective in picking up the geometric and textural coherence of face structures. FID provides distribution-level difference between real and generated images that takes into account realism as much as diversity. The two measures together provide an informed quantitative measure that quantifies fidelity and perceptual quality and is consistent with realistic face synthesis visual goals.

While strong, they are far from perfect. PSNR is artifact-free perceptually, and SSIM is contrast-sensitive; thus the incorporation of human judgment of perception to render it perfect.

CHAPTER FOUR

4. PROPOSED ARCHITECTURE

4.1 Overview of the Chapter

This chapter gives an overview description of an architecture of the algorithm proposed for color face generation, based on a combination of Mask R-CNN and GAN. The chapter has graphical figures as well as mathematical equations. The chapter is organized into four sections: the first section gives an overview description of the model architecture, and subsequently the second section gives a description of the generator and discriminator architecture. Part three describes the learning functions of the model, and the final section describes the training pseudocode.

4.2 Model Architecture

The model under proposal is a GAN-based face model for producing high-fidelity photo-realistic color facial images from grayscale sketches with semantic facial areas supervision and attention-augmented feature learning. The method begins with an inputting grayscale face sketch, which is passed to a pre-trained Mask R-CNN to produce a semantic segmentation mask that accurately describes main facial features such as eyes, nose, mouth, hair, and skin regions. The mask provides precise structural information that is blended with the sketch and sent to the generator to guide the synthesis process towards the appropriate spatially and semantically balanced output. The generator is realized in the form of a U-Net encoder-decoder architecture employing symmetric skip connections for preserving spatial information at each layer. Channel Attention modules in the encoder adaptively adjust the most important channel of features, strengthening the representation of noteworthy facial features selectively prior to being sent to skip connections. Self-Attention modules in the decoder are employed to entail long-range relationships and represent non-local relationships between spatial regions such that the generator can produce globally coherent and contextually coherent outputs. The discriminator network is multi-scale PatchGAN with three concurrent discriminators (D1, D2, D3) working at successively lower resolutions of 256×256, 128×128, and 64×64. The multi-scale architecture allows local fine-grained texture and global big-picture facial appearance to be evaluated in a single pass, improving adversarial feedback quality. The training objective amalgamates four interconnected loss functions: Adversarial loss to force the generator to generate outputs which are perceptually

indistinguishable from actual images by all three discriminators, Perceptual loss computed by taking high-level feature activations of a pre-trained VGG network to enforce output images having the same semantic content and visual features as the ground truth, Image reconstruction loss attained as an L1 distance to enforce faithful reconstruction of low-level color and texture; and SSIM loss for computing structural similarity between comparison and generated images with emphasis on perceptual fidelity of structures and minimizing distortions. Both discriminator and multi-scale generators are trained in an alternating manner, enabling stable adversarial optimization. The semantic mask guidance, attention-driven feature refinement, and multi-scale adversarial learning are employed to produce color facial images that not only appear realistic but also structurally consistent to the target identity and face layout.

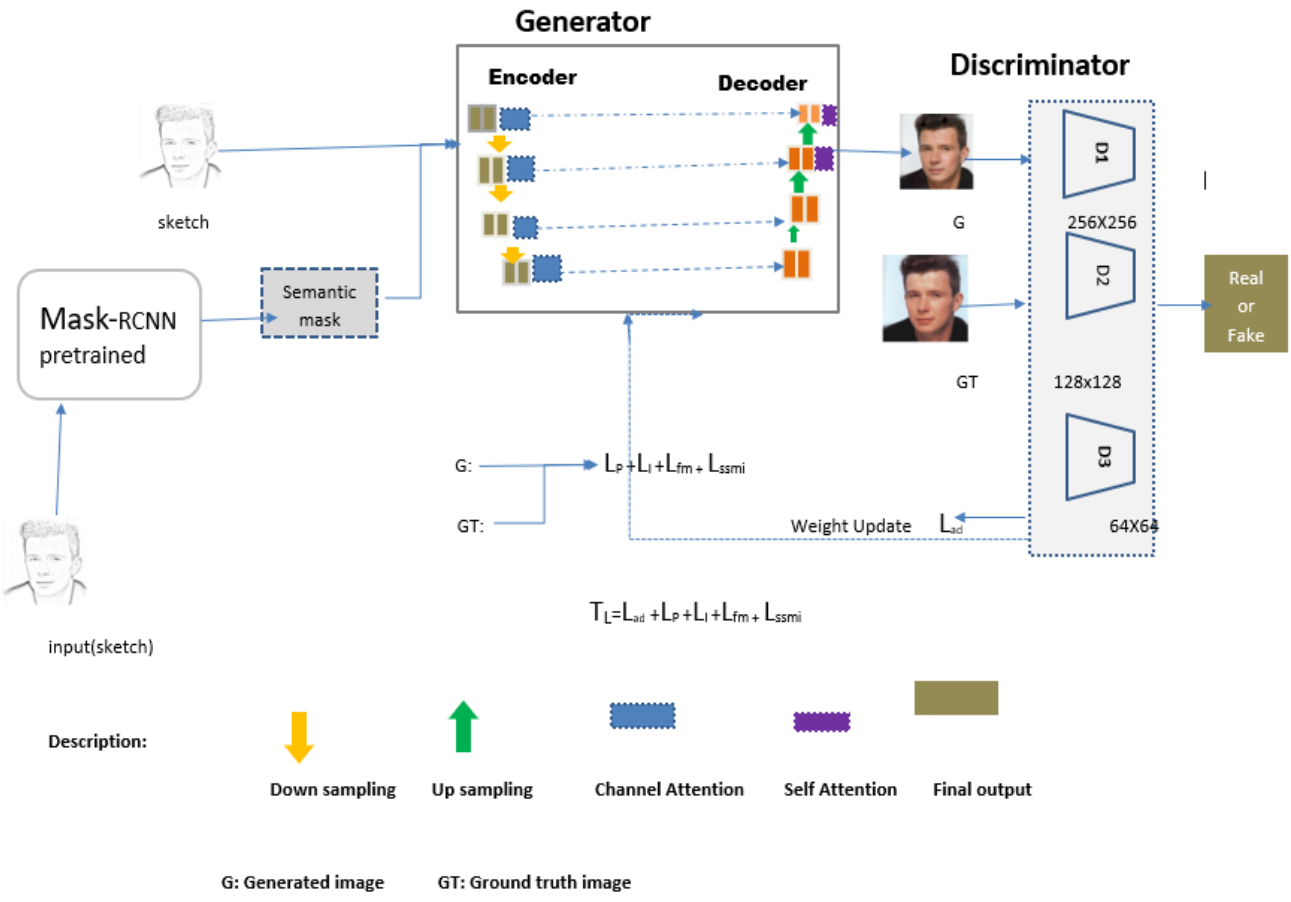


Figure 4. 1: Overall architecture of the proposed model

In our solution, we used channel attention for the encoder block and self-attention for the decoder block of the generator based on U-net in our model in the hope of building it stronger to generate a real facial image in the hope of deceiving the discriminator.

Self-attention in the GAN design allows the discriminator and generator to learn efficiently long-range relationships between image features (H. Zhang et al., 2018). In face generation, this plays a vital role in the sense that spatial relationships between elementary facial features like the eye, nose, and mouth are preserved. It also has high facial features like the skin texture, the boundary of the lip, and the acuity of the eye that helps it produce a more realistic facial image. Self-attention also lessens normal GAN fuzziness and deformation by giving a means to process and enhance synthesized features depending on the general context they form. In a bid to leverage these to their fullest, self-attention was integrated in the decoder part of the generator. Attention channel in deep convolutional neural networks has the responsibility of highlighting and pointing out key channels in a feature map. It enhances the performance of the network by reducing the processing noise and redundant information (Hu et al., 2018). It assists the model in paying greater attention to more important aspects of the face, such as shape and texture of eyes, mouth, and nose, thereby maintaining high-level facial image features. Channel attention was applied in the generator encoder module in this study. Adding channel attention to the encoder block enables one to establish a firm base for the rest of the layers in the model because the encoder diminishes major features through attention to the most significant features of the face, which are extremely important in precise generation.

4.3 Mask R-CNN Architecture

In the proposed model, a pre-trained Mask R-CNN (He et al. 2017) is employed as a critical semantic facial region detection tool. The model incorporates a ResNet-50 backbone and FPN and is pre-trained on COCO dataset (Lin et al. 2014) with instances of the human face annotated. This helps the system to correctly detect and contour major facial elements such as the eyes, nose, mouth, and jawline from grayscale input images. The produced binary facial masks draw attention to the semantically most important regions on the face and are appended to the input grayscale to provide a guided input to the generator. Blending gives spatial and semantic priors to direct the generator to focus on structurally meaningful areas and improve both the reality and coherence of produced color face images. Using a pre-trained Mask R-CNN has advantages: computational

expense is reduced, task-specific training of segmentation is bypassed, and excellent generalization and high face localization accuracy are obtained. The semantic guidance mechanism therefore facilitates the generator to maintain facial structure and detail more effectively in image generation.

4.4 Generator Architecture

In a GAN, the generator network is shown as the network that creates artificial data that is a perfect replica of the original data. It is usually constructed as a neural network, typically on top of convolution layers, that takes random noise as input and outputs data samples that are intended to be indistinguishable from real data. Our model employed a U-Net architecture (Ronneberger, Fischer, and Brox 2015) that is an encoder-decoder network with skip connection and hybrid attention, including self-attention, and channel-attention.

U-Net is a deep learning method originally suggested by, consisting of two primary paths: expansion and contraction. The expansion path consists of decoder layers that decode the processed data and add the contraction path information through skip connections to produce a segmentation map. Conversely, the contraction path takes a convolutional network structure with encoder blocks retaining contextual information and decreasing the spatial dimension of input (Ronneberger, Fischer, and Brox 2015). Convolution, Batch normalization, and ReLU constitute our encoder half of the generator. Transposed convolution, Batch normalization, and ReLU constitute the decoder half of the generator. For higher abstractions and conceptualized representations of the image, the bottleneck layer of the generator network receives the compressed form and lowers the dimensionality of the feature maps. The generator part of the network also applies skip connections to retain high-resolution features from earlier layers, something essential to retain detailed facial information through the process of transformation. Self-attention block is used at the third and fourth decoder layers in order to enable the generator to perceive global and local dependencies in mid-level feature maps. It can be used to help improve the spatial details and fine structures of the generated frontal faces. Channel attention is used at all the encoder layers in order to enable the network to concentrate on the most significant features.

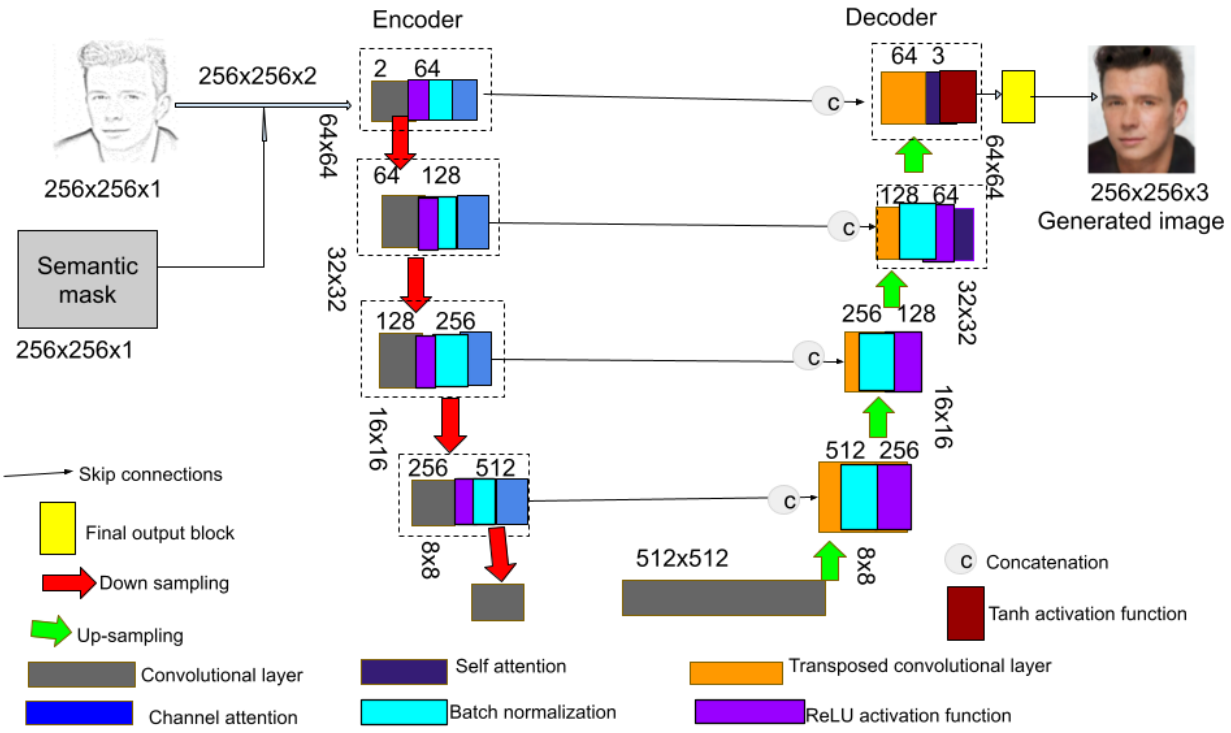


Figure 4. 2: Generator Architecture

In the architecture provided, the encoder block is a down-sampling module of four conv layers (conv1–conv4), with a channel attention block placed after each of them. Every layer of convolution is initialized with a stride = 2 and a kernel size = 4, followed by batch norm and ReLU activation. The decoder block is an up-sampling module of four deconv layers (deconv1–deconv4). The self-attention block is used in the third and fourth layers. Like the encoder, the kernel size of every deconvolutional layer is 4, and the stride is 2, and batch normalization and ReLU activation follow every layer but deconv4. In the final deconv4 layer, the Tanh activation function is used.

4.4.1 Channel Attention

Channel attention is initially introduced by SENet (Hu et al. 2020) to learn channel weights of each feature map. It is a mechanism applied in deep convolutional neural networks in a bid to specifically focus on the most relevant channels inside a feature map. It can render the network's performance enhanced through the reduction of the processing of the level of noise and unrelated

information. A channel attention map is created by applying feature inter-channel correlations to attend to significant areas in an input image by considering each channel of a feature map as a feature detector. The spatial dimension of the input feature map is down-scaled to compute channel attention in an efficient way by applying average pooling to pool spatial information (Woo et al. 2018). Attention channel blocks must selectively bring to the foreground significant informative feature channels and hide lesser ones.

They do this using channel-wise attention weights that modulate the feature maps. This way, the network can attend to the most discriminative features of the task. One computation of attention weights is typically part of a channel attention block. This typically consists of global average pooling followed by fully connected or convolutional layers in order to represent inter-channel dependencies. These attention weights are subsequently used to modulate the early feature maps, which will highlight relevant features and inhibit noise or irrelevant channels.

The provided equation defines a function that computes a channel-based statistic for a given input tensor. The input tensor is expected to be a 3D NumPy array with dimensions (K, H, W), where: K is the number of feature maps; H is the height of the feature maps, and W is the width of the feature maps

$$Z_c = HGP(y_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} y_c\,(i,j) \qquad (4.1)$$

Where:

- ✓ $y_c(i, j)$ is the value at spatial location (i, j) in channel c
- ✓ H and W are the height and width of the feature map y_c,
- ✓ *HGP denotes Global* Pooling, specifically average pooling over the spatial dimensions.

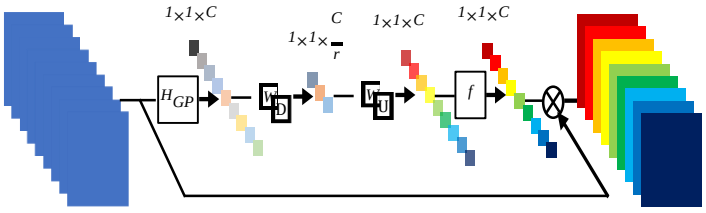


Figure 4.3: Channel attention (CA). ⊗ denotes element-wise product

4.4.2 Self-attention

Self-attention is a machine learning method, specifically applied to tasks like NLP. It helps the models understand the relationship between different elements of a sequence and how different elements of data

relate to each other. Later, this method was applied in many other tasks like computer vision. The majority of models for generating images are GANs constructed through convolutional layers. Since convolutional operation operates in a bounded neighborhood, it is computationally ineffective to solely employ convolutional layers for extracting long-range relations in images. SAGAN was introduced by (Zhang et al. 2019), which incorporated self-attention into the GAN framework in such a way that both generator and discriminator are able to extract long-range relations efficiently at low computational expense. SAGAN applies spectral normalization to maintain training computation cost low.

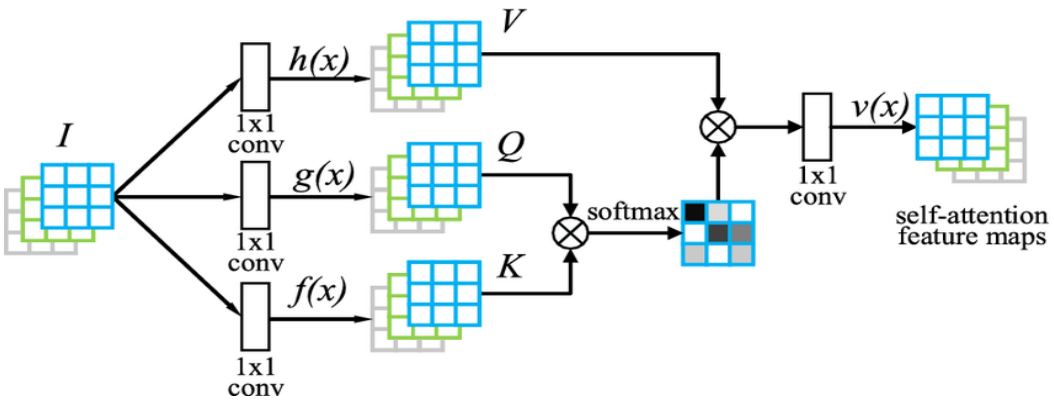


Figure 4. 3: Self-attention Architecture

Self-attention enables the generator to capture local and global dependencies within intermediate feature maps. This can help improve the spatial details and fine structures of the synthesized images.

4.5 Discriminator Architecture

It is a challenging task for GANs to generate high-resolution color face images since the discriminator needs to pay attention to both finely detailed facial features (e.g., hair texture, skin pores) and global structural consistency (e.g., face symmetry, face ratio) at the same time. To this end, we use a multi-scale discriminator design from prior high-resolution image generation work (Wang, Sindagi, and Patel, 2018). We employ three discriminators (D_1, D_2, and D_3) at different spatial resolutions in our model, where the input image is progressively down-sampled to form a multi-resolution pyramid. The largest discriminator (D_1) is applied for the lowest resolution, which encourages realistic high-frequency detail such as skin texture and fine color transitions (Karras,

Laine, and Aila 2021). The mid-scale (D_2) and small-scale (D_3) discriminators operate on lower-resolution images with the aim of supplying mid-level feature constancy (e.g., correct facial feature alignment) and global structural validity. This hierarchical discriminative method offers thrifty receptive field coverage without a dense network and computational efficiency without overfitting problems.

The adversarial objective is formulated as a multi-task optimization problem:

$$\min_{G} \max_{D_1,D_2,D_3} \sum_{k=1}^{3} \mathcal{L}_{GAN}(G, D_k),$$

Where all the discriminators collaborate towards training stabilization and synthesis quality enhancement (Miyato et al. 2018). The approach effectively removes most of the GAN artifacts in face generation, such as blurring, copy-pasting textures, and structural distortion. The multi-scale pipeline also supports progressive training, which allows for ongoing resolution scaling by introducing new discriminators at early, coarser resolutions without re-starting training from the beginning (Karras, Laine, and Aila 2019).

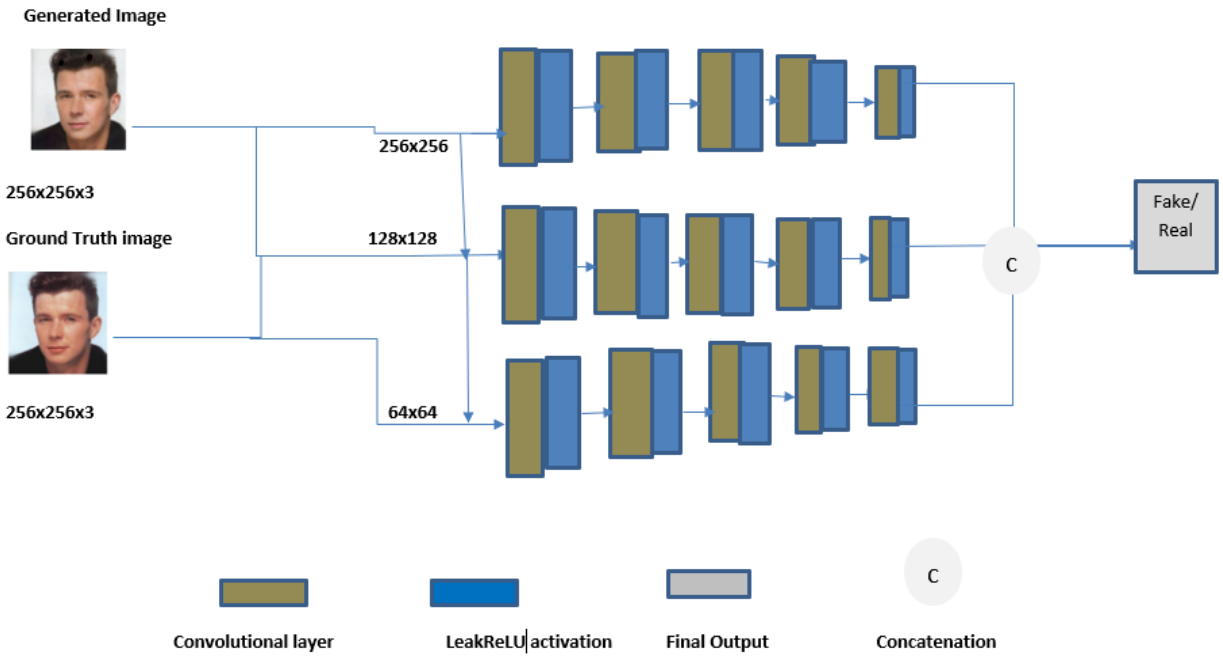


Figure 4. 5: Discriminator Architecture

4.6 Learning Function of the Model

4.6.1 Activation Functions

Activation functions of various kinds are employed in the various regions of the network to improve the efficiency of learning. Rectified Linear Unit (ReLU) activation function is applied in the encoder, decoder, pyramid, and context modules of the generator for eliminating the problem of vanishing gradients and facilitating effective feature learning. Hyperbolic tangent function (Tanh), however, is the output activation function and effectively normalizes pixel values in the interval [−1, 1] for facilitating normally-ordered image datasets. In the discriminator component, Leaky ReLU (with α=0.2) is applied in the convolution layers for eliminating the problem of neuron inactivation and allowing a minimum flow of gradients for negative input values, and the final layer applies the Sigmoid function for outputting probabilities supportive of binary classification (real or fake).

4.6.2 Loss Functions

Planning a suitable learning function is thought to be quite important for producing better results. Adversarial loss, Perceptual loss (perc-loss), Feature matching loss, and LPIPS loss are used in this work:

Adversarial loss (ad-loss) is one of the components of GANs, introduced by Goodfellow et al. (Goodfellow et al. 2014). The loss function establishes a minimax game between a generator and a discriminator, where the goal of the generator is to produce realistic samples and the discriminator tries to distinguish between real and generated data. The original version employs a binary cross-entropy loss, but later refinements such as Wasserstein GAN (Arjovsky, Chintala, and Bottou 2017) and LSGAN (Mao et al. 2017) have suggested more stable alternatives. The adversarial loss is very good at producing sharp and realistic images, yet it can be challenging to train because of instability problems.

Perceptual loss (perc-loss), proposed by Johnson et al. (Johnson, Alahi, and Fei-Fei 2016), was the turning point in image generation tasks by breaking from pixel-wise comparisons. Instead of calculating pixel-wise differences, this loss calculates deep feature differences using pretrained networks (most commonly VGG). The intuition here is that deep feature comparison preserves semantic content better than pixel-wise losses and is particularly useful for style transfer and super-

resolution applications. Loss calculates the L2 distance between intermediate activations of target and synthesized images and therefore encourages the network to produce semantically similar outputs rather than just pixel-perfect outputs

Pixel-wise Loss (L1) measures the average pixel-level difference between generated and target images. Lower L1 shows better reconstruction of facial structures.

The mathematical formulation is:

$$\mathcal{L}_{L1}(G) = \frac{1}{N}\sum_{i=1}^{N} |G(x)_i - y_i| \qquad \text{(eq. 4.3)}$$

Where:

$N \rightarrow$ Total number of pixels in the image

$G(x)_i \rightarrow$ Pixel value of the generated image at position i

$y_i \rightarrow$ Pixel value of the ground truth image at position i

Feature matching loss, proposed by Salimans et al. (2016), ameliorates training instability by requesting the generator to match real data statistics in the discriminator's feature space. Instead of optimizing the adversarial objective explicitly, this loss minimizes the L1 difference between intermediate discriminator features for the fake and real images. This technique provides stronger gradients during training and is now a standard technique in modern GAN models like ProGAN and StyleGAN, and it avoids mode collapse and improves convergence.

$$L_{\text{FM}} = E_{x \sim p_{\text{data}}}\left[\sum_{l} \| D^{(l)}(x) - D^{(l)}(G(z)) \|_1\right] \qquad (eq\ 4.4)$$

Where

- ✓ $x \sim p_{\text{data}}$: A real data sample from the dataset.
- ✓ z: Input to the generator.
- ✓ $G(z)$: The generated image.
- ✓ $D^{(l)}(x)$: Feature map at the l^{th}lth layer of the discriminator for real input.
- ✓ $D^{(l)}(G(z))$: Feature map at the l^{th}lth layer for the generated image.

- ✓ $\|\cdot\|_1$: The L1 norm between feature maps

Structural Similarity Index Measure (SSIM) Loss, introduced by Wang et al. (2004), is a human visual perception-correlated perceptual measure for image similarity comparison. In contrast to the traditional pixel-level loss functions such as L1 or L2, which define the absolute difference between the intensity of pixels, SSIM computes similarity from luminance, contrast, and structural information. Its scale is between -1 and 1, and a value of 1 indicates a completely identical structure in the images being compared.

The SSIM metric is formulated as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \qquad \text{(eq 4.5)}$$

Where:

- ✓ μ_x, μ_y are the mean intensities of the two images.
- ✓ σ_x, σ_y are the variances, representing contrast.
- ✓ σ_{xy} is the covariance, representing structural similarity.
- ✓ C_1 and C_2 are small constants to avoid division by zero.

SSIM loss is defined as:

$$\mathcal{L}_{\text{SSIM}} = 1 - \text{SSIM}(x, y) \qquad \text{(eq 4.6)}$$

Where:

$\text{SSIM}(x, y)$ →Structural similarity index between the generated image x and ground truth y.

$\mathcal{L}_{\text{SSIM}}$ → Ranges from 0 (perfect similarity) to 1 (completely dissimilar).

4.7 Training Procedure Pseudocode

The following pseudocode describes the training process for the proposed model:

ALGORITHM: PSEUDOCODE FOR PROPOSED MODEL

1. *Initialize the weights of Generator G and Multiscale Discriminator D*
   - *Channel attention in the encoder and self-attention in the decoder of the U-Net-based generator*
   - *Employ a pre-trained Mask R-CNN to extract semantic facial masks*
   - *Use a multiscale discriminator to enhance adversarial learning at multiple resolutions*

2. Input: Sketch face image I_g, Ground truth color image I_{rgb}

3. *For each training epoch, do*

   For each batch of training samples, do

   A. *Extract face region mask from I_g using Mask R-CNN*

   B. *Concatenate I_g and mask M to form the generator input I_{input}*

   C. *Generate color face image:*

$$\hat{I}_{rgb} = G(I_{input})$$

   D. *Update Multiscale Discriminator D using adversarial loss:*

$$L_D = \sum_{s \in S} \left[E_{x \sim p_{real}}[log D_s(x)] + E_{z \sim p_{gen}}\left[log\left(1 - D_s(G(z))\right)\right] \right]$$

   E. *Update Generator $using$ the combined loss:*

$$L_G = \lambda_1 L_{adv} + \lambda_2 L_{perc} + \lambda_3 L_{fm}$$

   L_{adv}: Adversarial loss

   L_{perc}: Perceptual loss using VGG-based feature similarity

   L_{fm}: Feature matching loss from intermediate discriminator layers

4. *Save the generator and discriminator model weights*
5. *End loop*
6. *Final output* : $Colored\ face\ image\ \hat{I}_{rgb}$

CHAPTER FIVE

5. IMPLEMENTATION AND EXPERIMENTS

5.1 Chapter Overview

This chapter explains the execution of the proposed model, including the working environment, data augmentation procedures, the application of color face generation, and the experimental procedures adopted.

5.2 Working Environment

5.2.1 Hardware Components

Here, the hardware tool configuration is explained that was utilized to execute the model in the environment.

Laptop:

- ✓ Processor: Intel(R) Core (TM) i5-4210M CPU @ 2.60GHz 2.60 GHz
- ✓ RAM: 8.00 GB
- ✓ Operating system: Windows 10
- ✓ System type: 64-bit, x64-based processor

Desktop:

- ✓ Processor: Intel(R) Core (TM) i7-8700M CPU @ 3.20GHz 3.19 GHz
- ✓ Installed RAM: 8.00 GB
- ✓ Operating system: Windows 10
- ✓ System type: 64-bit, x64-based processor
- ✓ Graphics: Intel ® HD Graphics 520/ NVIDIA Corporation
- ✓ GPU: GeForce RTX 2070 Super 8 GB RAM

5.2.2 Software Components

The software environment included a combination of programming languages, frameworks, and cloud-based platforms to facilitate model development and experimentation:

- ✓ Python: A high-level, general-purpose programming language widely used in machine learning and data science.
- ✓ PyTorch: An open-source deep learning framework providing efficient machine learning and mathematical computation utilities.

- ✓ Keras: A high-level API over TensorFlow, which enables quick and flexible construction and training of deep learning models.
- ✓ Google Drive: Used for cloud storage, dataset upkeep, and Google Colab integration during preprocessing and training.
- ✓ Google Colab: Cloud-based, free Jupyter Notebook environment with GPU resources for training models without local installation.
- ✓ Kaggle: Cloud-based data science environment with pre-installed Python libraries, free NVIDIA Tesla P100 GPUs, and Google Cloud TPUs for accelerated deep learning model training. This arrangement ensured reproducibility, scalability, and reduced reliance on local hardware.
- ✓ Anaconda: A Python distribution used for efficient package and virtual environment management in data science and machine learning processes.

5.3 Training Procedure

To meet the goals of this research, the following procedures were carried out:

- ✓ Data Collection: Gathered CelebA-HQ face image datasets from public archives.
- ✓ Data Preprocessing: Resized all images to standard dimensions appropriate for model input (256x256 pixels). Normalized pixel values to the range of [-1, 1] to accommodate the requirements of both the Mask R-CNN and GAN models, as well as Data Augmentation.
- ✓ GPU Configuration: -Configure an NVIDIA GPU server to maximize computation effectiveness
- ✓ Deployment of the Proposed Solution: Developed the hybrid model that combines the advantage of Mask R-CNN in semantic facial mask extraction and GAN in face image synthesis with color.

5.4 Dataset Preparation

5.4.1 Mask R-CNN Implementation

Before dividing the dataset, a pre-trained Mask R-CNN was utilized to create semantic masks for segmentation on each of the images. Mask R-CNN is a deep learning model designed for instance-level object detection and segmentation, and in this study, it was used to accurately segment salient

facial regions, including skin, eyes, mouth, hair, and accessories. The architecture integrates a backbone feature extractor (ResNet and FPN) to carry out hierarchical feature extraction, a Region Proposal Network (RPN) to detect candidate object regions, and segmentation heads to produce pixel-level masks for each facial component that has been detected. The masks are used as spatial guidance to the GAN so that realistic and structurally consistent face images are produced. Leveraging a pretrained Mask R-CNN allowed the study to utilize existing segmentation learning that reduced training time and improved accuracy in the CelebA-HQ dataset.

A sample implementation of the Mask R-CNN code used in this study is shown below:

```
# Configure pretrained Mask R-CNN
cfg = get_cfg()
cfg.merge_from_file(model_zoo.get_config_file(
    "COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
cfg.MODEL.ROI_HEADS.NUM_CLASSES = len(FACIAL_CLASSES)
cfg.MODEL.WEIGHTS = "/kaggle/input/mask-model/model_final.pth"
cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.4
cfg.MODEL.DEVICE = "cuda" if torch.cuda.is_available() else "cpu"
predictor = DefaultPredictor(cfg)

# Inference and semantic mask generation
for img_path in sorted(IMG_DIR.glob("*.jpg")):
    img = cv2.imread(str(img_path))
    outputs = predictor(img)
    inst = outputs["instances"].to("cpu")

    sem_map = np.zeros(img.shape[:2], dtype=np.uint8)
    for mask, cid, score in zip(inst.pred_masks, inst.pred_classes, inst.scores):
        if score >= cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST:
            sem_map[mask.numpy()] = cid + 1

    cv2.imwrite(str(OUT_SEM / f"{img_path.stem}_semantic.png"), sem_map)
```

Sample code 5.1: Sample Mask R-CNN code for semantic mask generation.

5.4.2 Dataset Splitting and Loading

The dataset is divided into three sets: training (70%), testing (15%), and validation (15%). To prevent leakage of data, the test set is isolated from the overall dataset first, and then the remaining data is divided into training and validation sets. All the images are normalized by resizing all the images to 256 × 256 pixels, which gives a uniform input size, improves computational efficiency, and reduces training time. Also, pixel values are normalized from their original range of [0, 255]

to [0, 1]. Normalization enhances numerical stability, accelerates gradient-based optimization, enables color channels to be scaled uniformly, and benefits general model performance.

```
# Dataset loading and splitting
    dataset = GANInputDataset("/Kaggle/input/sketches", "/kaggle/input/mask-
output1/semantic",
 "/kaggle/input/target")
    total = len(dataset)
    train_size = int (0.7*total)
    val_size = int (0.15*total)
    test_size = total - train_size - val_size
    train_set, val_set, test_set = torch.utils.data.random_split(dataset,
[train_size, val_size, test_size])
    # Apply transforms
    train_set.dataset.transform = train_transform
    val_set.dataset.transform = val_transform
    train_loader = DataLoader(train_set, batch_size=config["batch_size"],
shuffle=True, num_workers=2)
    val_loader = DataLoader(val_set, batch_size=config["batch_size"],
shuffle=False, num_workers=2)
test_loader = DataLoader(test_set, batch_size=config["batch_size"],
shuffle=False, num_workers=2)
```

Sample code 5.2: Dataset Loading and Splitting

5.5 Dataset Preprocessing Implementation

As stated above, different data preprocessing techniques were utilized in this research that focused on properly preparing the training images and enhancing the performance of the model. Data augmentation methods were also utilized to generate diverse forms of the images.

```
# Training Datasets augmentations
train_transform = transforms.Compose([
    transforms.Resize((size, size)),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.RandomCrop(size, padding=10),
    transforms.ToTensor(),
    transforms.Normalize([0.5]*3, [0.5]*3)
])
# Training augmentations
train_transform = transforms.Compose([
    transforms.Resize((size, size)),
    transforms.RandomHorizontalFlip(p=0.5),
    transforms.RandomCrop(size, padding=10),
    transforms.ToTensor(),
    transforms.Normalize([0.5]*3, [0.5]*3)
])
```

Sample code 5.3: Dataset augmentation

5.6 Implementation of Proposed Model

The proposed model is implemented using the PyTorch deep learning framework, benefiting from its dynamic computation graph, GPU acceleration, and a large set of built-in modules and pre-trained models. There are three key components in the architecture: a pre-trained Mask R-CNN for semantic facial region extraction, an attention-augmented U-Net generator, and a Patch-based discriminator with spectral normalization and feature matching. In the following, each module and its implementation are described in detail.

The input image for every sketch is preprocessed, normalizing the pixel values to [0, 1]. A pre-trained Mask R-CNN model (He et al. 2017), with ResNet-50 backbone and FPN, is used to obtain facial segmentation masks. The masks highlight significant facial features such as eyes, nose, mouth, and skin contours. The binary mask is then concatenated as a second channel alongside the grayscale image to create a 2-channel input tensor that guides the generator's attention by putting emphasis on semantically significant facial regions. This semantic and spatial guidance causes the generator to focus its efforts on areas that are of highest relevance to human perception of face realism.

5.6.1 Implementation of Encoder Block

In the proposed hybrid GAN model, the encoder block of the generator is responsible for extracting hierarchical features from input images while progressively down-sampling their spatial resolution. The encoder has four convolutional layers whose explicit aim is to down-sample the input progressively in spatial resolution and up-sample it in depth of feature representation. All the convolution layers have the same kernel of 4×4, a stride of 2, and a padding of 1 for effective down-sampling and feature learning. Instance Normalization is done after each convolution to normalize feature maps to stabilize the network training, especially when using small batch sizes. A LeakyReLU activation layer with a negative slope of 0.2 is added after normalizing to introduce non-linearity and allow the network to learn sophisticated mappings. It also avails the network against the vanishing gradient problem, resulting in better feature propagation. Channel attention modules are added before each encoder block to continue guiding the attention of the network towards more informative channels about facial structures.

```
# Encoder Block 1
self.enc1 = nn.Sequential(
    nn.Conv2d(in_channels, base, kernel_size=4, stride=2, padding=1),
    nn.InstanceNorm2d(base),
    nn.LeakyReLU(0.2, inplace=True),
    ResidualBlock(base),
    ChannelAttention(base)  # Channel Attention applied here
)
# Encoder Block 2
self.enc2 = nn.Sequential(
    nn.Conv2d(base, base*2, kernel_size=4, stride=2, padding=1),
    nn.InstanceNorm2d(base*2),
    nn.LeakyReLU(0.2, inplace=True),
    ResidualBlock(base*2),
    ChannelAttention(base*2)  # Channel Attention applied here
)
# Encoder Block 3
self.enc3 = nn.Sequential(
    nn.Conv2d(base*2, base*4, kernel_size=4, stride=2, padding=1),
    nn.InstanceNorm2d(base*4),
    nn.LeakyReLU(0.2, inplace=True),
    ResidualBlock(base*4),
    ChannelAttention(base*4)  # Channel Attention applied here
)
# Encoder Block 4
self.enc4 = nn.Sequential(
    nn.Conv2d(base*4, base*8, kernel_size=4, stride=2, padding=1),
    nn.InstanceNorm2d(base*8),
    nn.LeakyReLU(0.2, inplace=True),
    ResidualBlock(base*8),
    ChannelAttention(base*8)  # Channel Attention applied here
)
```

Sample code 5.4: Implementation of Encoder Block

5.6.2 Bottleneck Layer Implementation

The bottleneck layer serves as the core feature transformation module of the generator network. The bottleneck begins with a 3×3 padded convolutional layer to preserve spatial resolution, followed by Instance Normalization and a ReLU activation, which enables stable and effective training. This is followed by a residual block, made of two convolutional layers, each followed by Instance Normalization and ReLU activations, to facilitate improved gradient flow and feature enhancement using skip connections. Placed between residual blocks, a self-attention module extracts long-range spatial dependency to enable the network to encode global contextual information that is important in synthesizing coherent facial features. Synthesizing with this, a channel attention module is able to dynamically recalibrate channel-wise feature responses adaptively for better discriminative feature representation. All these modules together enable the

bottleneck to transform encoded features into a rich, globally-informed representation and optimize them for subsequent decoding and reconstruction stages.

```
self.bottleneck = nn.Sequential(
    # 3x3 Conv to keep spatial resolution
    nn.Conv2d(base * 8, base * 8, kernel_size=3, padding=1),
    # Normalization to stabilize training
    nn.InstanceNorm2d(base * 8),
    # Non-linear activation
    nn.ReLU(inplace=True),
    # Residual Block for better gradient flow and feature refinement
    ResidualBlock(base * 8),
    # Self-Attention layer to capture long-range spatial dependencies
    SelfAttention(base * 8),

    # Channel Attention to recalibrate channel-wise feature responses
    ChannelAttention(base * 8)
)
```

Sample code 5.5: Implementation of Bottleneck Layer

5.6.3 Decoder Block Implementation

The decoder restores feature representation-based high-resolution photorealistic color facial images from the bottleneck and the encoder. The decoder consists of four blocks, each of which starts with an up-sampling operation (in general, bilinear interpolation with a factor of 2), followed by a 3×3 convolutional layer. Instance Normalization and LeakyReLU activation are used after every convolution to stabilize and activate the output. Other than that, each block has a residual block to continue refining the features and bringing consistency to spatial information. Skip connections that are part of the corresponding encoder layers are concatenated in each step of the decoder, which preserves low-level spatial features of value to correctly reconstruct. The final output layer consists of a 3×3 convolutional layer to project the features into a 3-channel RGB image and subsequently apply Tanh activation to output pixel values in the normalized range of [-1, 1]. The self-attention introduced so far in the bottleneck allows the decoding process to maintain global coherence and local detail.

```
self.up4 = nn.ConvTranspose2d(base*8, base*4, kernel_size=2, stride=2)
self.dec4 = nn.Sequential(
    nn.Conv2d(base*8 + base*4, base*4, kernel_size=3, padding=1),
    nn.InstanceNorm2d(base*4),
    nn.ReLU(inplace=True),
    ResidualBlock(base*4),
    SelfAttention(base*4)    # Self Attention applied here
)
self.up3 = nn.ConvTranspose2d(base*4, base*2, kernel_size=2, stride=2)
self.dec3 = nn.Sequential(
    nn.Conv2d(base*4 + base*2, base*2, kernel_size=3, padding=1),
    nn.InstanceNorm2d(base*2),
    nn.ReLU(inplace=True),
    ResidualBlock(base*2),
    SelfAttention(base*2)    # Self Attention applied here
)

self.up2 = nn.ConvTranspose2d(base*2, base, kernel_size=2, stride=2)
self.dec2 = nn.Sequential(
    nn.Conv2d(base*2 + base, base, kernel_size=3, padding=1),
    nn.InstanceNorm2d(base),
    nn.ReLU(inplace=True),
    ResidualBlock(base)

)
self.up1 = nn.ConvTranspose2d(base, base//2, kernel_size=2, stride=2)
self.dec1 = nn.Sequential(
    nn.Conv2d(base + base//2, base//2, kernel_size=3, padding=1),
    nn.InstanceNorm2d(base//2),
    nn.ReLU(inplace=True),
    ResidualBlock(base//2)
)
 # Final Output Layer
 self.final = nn.Sequential(
     nn.Conv2d(base//2, out_channels, kernel_size=3, padding=1),
     nn.Tanh()
 )
```

Sample code 5.6: Implementation Decoder Block

5.6.4 Implementation of Multiscale Discriminator

A multiscale discriminator network used in the current GAN is composed of multiple PatchGAN discriminators at different resolutions of an input image. The model is thus able to gain and process image details at different scales. Local fine texture, medium-level structure, and global coherence are attained by full, half, and quarter resolution processing of the input image, respectively. All the discriminators utilize the same five convolutional block architecture. The initial block employs 4×4 convolution with a stride of 2 and 64 filters to extract low-level input features and applies the LeakyReLU activation function of slope 0.2. The initial block does not apply normalization to

make raw input data available. The next blocks doubled the number of filters to 128, 256, and 512 with 4×4 convolutions, 2 or 1 strides, padding 1, and Instance Normalization following every convolution in order to stabilize training and enhance texture and style consistency. LeakyReLU activations are used following every normalization layer to cause feature propagation. The last convolutional layer uses a 4×4 kernel with a stride of 1 and a pad of 1 to provide a one-channel patch-level real/fake prediction map. Down-sampling the input image progressively for the second and third discriminators, the multiscale architecture necessitates that the GAN produce facial images of high fidelity and structural coherence at a large number of spatial scales.

```
# Multiscale Discriminator with 3 scales
class MultiScaleDiscriminator(nn.Module):
    def __init__(self, num_discriminators=3):
        super().__init__()
        self.num_discriminators = num_discriminators
        self.discriminators = nn.ModuleList()
        for _ in range(num_discriminators):

self.discriminators.append(SingleScaleDiscriminator())

    def forward(self, x):
        results = []
        input_downsampled = x
        for i, D in enumerate(self.discriminators):
            out = D(input_downsampled)
            results.append(out)
            if i != self.num_discriminators - 1:
              # Downsample input for next discriminator (by 2x)
                  input_downsampled =
F.avg_pool2d(input_downsampled, kernel_size=3, stride=2,
padding=1)
        return results
```

Sample code 5.7: Implementation of Multiscale Discriminator

Self-attention is utilized in the bottleneck and selected decoder blocks in the proposed generator to learn long-distance spatial relationships between feature maps. Unlike standard convolutional layers that are limited to knowing just local neighborhoods, self-attention enables the network to know far-away spatial locations and preserve global facial feature consistency. The mechanism transforms the input feature map to query, key, and value tensors through 1×1 convolution. Attention scores are computed as the dot product of keys and queries with SoftMax normalization to produce a spatial attention map, which is applied to reweight the value tensor according to relevance. A residual connection and a learnable scaling factor are employed to blend the attention-augmented output with the original input, preserving local details while fusing global context. This

technique allows the generator to have consistent relations between significant facial regions, i.e., the eyes, nose, and mouth, to produce more realistic appearances and visually consistent color face images.

```
class SelfAttentionBlock(nn.Module):
    def __init__(self, in_channels):
        super().__init__()
        self.query = nn.Conv2d(in_channels, in_channels//8, 1)
        self.key   = nn.Conv2d(in_channels, in_channels//8, 1)
        self.value = nn.Conv2d(in_channels, in_channels, 1)
        self.gamma = nn.Parameter(torch.zeros(1))
        self.softmax = nn.Softmax(dim=-1)
    def forward(self, x):
        B, C, H, W = x.shape
        q = self.query(x).view(B, -1, H*W).permute(0,2,1)
        k = self.key(x).view(B, -1, H*W)
        attn = self.softmax(torch.bmm(q,k))
        v = self.value(x).view(B, -1, H*W)
        out = torch.bmm(v, attn.permute(0,2,1)).view(B,C,H,W)
        return self.gamma*out + x
```

Sample code 5.8: Implementation of self-attention

In the proposed generation model, channel attention is employed in the encoder, bottleneck, and selected decoder blocks to dynamically rebalance the importance of the feature channels. In comparison with self-attention, which is spatial relationship-oriented, channel attention is targeted to the most informative channels to encourage discriminative feature representation. The application first performs global average pooling to convert each channel into a descriptor, with the entire channel response being extracted in the spatial dimensions. This descriptor passes through a very tight bottleneck of fully connected non-linear layers to generate channel-wise weights between 0 and 1 using a sigmoid function. The weights are subsequently applied to down-weight the original feature map in such a way that channels with more information are highlighted, while less informative ones are suppressed. Through attention direction, the network enforces its attention to meaningful face parts such as eyes, nose, and mouth, and produces more realistic and structure-consistent synthesized color face images.

```python
# Attention modules
class ChannelAttention(nn.Module):
    def __init__(self, in_channels, reduction=16):
        super().__init__()
        self.avg_pool = nn.AdaptiveAvgPool2d(1)
        self.fc = nn.Sequential(
            nn.Linear(in_channels, in_channels // reduction, bias=False),
            nn.ReLU(inplace=True),
            nn.Linear(in_channels // reduction, in_channels, bias=False),
            nn.Sigmoid()
        )
    def forward(self, x):
        b, c, _, _ = x.size()
        y = self.avg_pool(x).view(b, c)
        y = self.fc(y).view(b, c, 1, 1)
        return x * y.expand_as(x)
```

Sample code 5.9: Implementation of channel attention

In the proposed GAN model, adversarial loss is formulated with a hinge loss function, which stabilizes training and improves the color face image quality generated. For the discriminator, hinge loss promotes predictions on the real images that are more than a margin of one and predictions on the generated images that are less than a margin of negative one, effectively pushing the discriminator to assign high scores to real images and low scores to generated images. For the generator, the adversarial hinge loss is the negative mean of discriminator predictions on generated images, which causes the generator to produce images that are as real as possible for the discriminator. This causes the generator to learn to produce photorealistic facial images that can effectively fool the discriminator, and the discriminator keeps improving its discrimination ability for real and synthesized sample discrimination. With the fusion of the adversarial loss and other reconstruction, perceptual, feature-matching, and structural similarity losses, the model aims to achieve a balance in fidelity, realism, and global coherence of the color face images generated.

```python
def discriminator_hinge_loss(real_pred, fake_pred):
    loss_real = torch.mean(F.relu(1.0 - real_pred))
    loss_fake = torch.mean(F.relu(1.0 + fake_pred))
    return 0.5 * (loss_real + loss_fake)

def generator_hinge_loss(fake_pred):
    return -torch.mean(fake_pred)
```

Sample code 5.10: Implementation of Adversarial loss

In the proposed GAN model, perceptual loss is used to not only render the generated color face images pixel-wise similar to the ground truth but also to preserve high-level semantic and

perceptual features. It achieves this by passing the generated and real images through a pre-trained VGG19 network and fusing features of an intermediate layer. The perceptual loss is then taken to be the L1 distance between such feature maps, which penalizes differences in content and texture rather than raw pixel values. Before feature extraction, normalization is applied to match the conditions of the pretraining dataset using the standard mean and standard deviation of the VGG network. By minimizing this loss, the generator is incentivized to produce images that are perceptually similar to real faces, with fine details, textures, and overall facial structures intact that are important to realistic synthesis. Pixel-wise L1 and adversarial losses are complemented by this loss by emphasizing higher-level feature constancy, which enhances visual fidelity in output images.

```
class VGGPerceptualLoss(nn.Module):
    def __init__(self, device):
        super().__init__()
        vgg = models.vgg16(weights=models.VGG16_Weights.IMAGENET1K_V1).features[:16].to(device)
        for p in vgg.parameters():
            p.requires_grad=False
        self.vgg = vgg
        self.mean = torch.tensor([0.485,0.456,0.406]).view(1,3,1,1).to(device)
        self.std  = torch.tensor([0.229,0.224,0.225]).view(1,3,1,1).to(device)

    def forward(self, input, target):
        input = (input + 1)/2
        target = (target + 1)/2
        input = (input - self.mean)/self.std
        target = (target - self.mean)/self.std
        return F.l1_loss(self.vgg(input), self.vgg(target))
```

Sample code 5.11: Implementation of Perceptual loss

In addition to perceptual and adversarial losses, a feature-matching loss is proposed to regularize GAN training and encourage synthesis of high-frequency details. The loss measures the L1 distance of the discriminator intermediate feature maps between synthesized and real images. By compelling the generator to match internal representations of real images, the network is instructed to produce more isotropic textures, details, and structural patterns. In comparison to adversarial loss, which merely penalizes the generator based on the final real/fake label of the discriminator, feature-matching loss gives explicit supervision to middle-level feature distributions, leading to smoother convergence and improved visual quality of synthetic color face images.

```python
def feature_matching_loss(D, real, fake):
    loss = 0
    # Extract intermediate features (all layers except the last output layer)
    real_feats = D.model[:-1](real)
    fake_feats = D.model[:-1](fake)
    loss += F.l1_loss(fake_feats, real_feats)
    return loss
```

Sample code 5.12: Implementation of Feature matching loss

To maintain spatial structures and local relations of facial structures, a structural similarity loss (SSIM loss) is introduced. SSIM calculates luminance, contrast, and structural correlations between the synthesized and ground-truth images. Through maximizing the SSIM score (or equivalently minimizing 1–SSIM), the generator is encouraged to maintain consistent textures, edges, and shapes, which are essential for realistic face synthesis. This loss is particularly effective for maintaining high-frequency details and preventing blurring, and it is complementary to pixel-wise L1 loss and perceptual loss for high-fidelity image synthesis.

```python
from torchmetrics.image import StructuralSimilarityIndexMeasure

ssim_loss_fn = StructuralSimilarityIndexMeasure(data_range=1.0).to(device)

# Usage
ssim_value = ssim_loss_fn(fake_img, real_img)
ssim_loss = 1 - ssim_value  # minimizing (1 - SSIM) as a loss
```

Sample code 5.13: Implementation of Structural similarity loss

5.7 Hyperparameter Configuration

Hyperparameters are settings used to control the learning process of a machine learning model. These are tunable variables and affect the model training performance directly. Optimizers are mechanisms that regulate a neural network's parameters when training to minimize the error or loss function. The Adam optimizer was used to update the network weights with a learning rate of 0.0002 for the generator and 0.0001 for the discriminator, and momentum parameters set to beta_1=0.5 and beta_2=0.999. Adam is rooted in two of the most popular optimization methods, AdaGrad and RMSProp (Kingma & Ba, 2014). The algorithm supports automatic learning rate adjustment for every parameter separately and is characterized by relative ease of application with minimal hyperparameter adjustment possibilities. The learning rate regulates update sizes during

the process of optimization and controls the extent to which model parameters are updated, considering the loss function's gradient. We had originally trained the model with a batch size of 32 for 100 epochs.

Table 5. 1: Hyperparameters Configuration

Hyperparameters	***Values***
Learning Rate (Generator)	0.0002
Learning Rate (Discriminator)	0.0001
Batch Size	32
Number of Epochs	100
Optimizer	Adam

Table 5.5 shows the hyperparameters for training our model. GAN training must be computationally costly; therefore, the use of large batch sizes is limited by hardware. Conversely, extremely small batch sizes have the drawback of greater overall training time. Therefore, we employed a moderately-sized batch size of 32 in trying to strike a balance between computational cost and training performance. We used the Adam optimizer rather than its variants, like SGD, because of its property of adapting the learning rate for individual parameters. This flexibility can be particularly useful in GANs where parameters are converging differently, utilized to stabilize training and improve convergence.

5.8 Experiment Class

Table 5. 2: List of experiment classes

Experiments	***Models***	***Dataset***
1	Baseline GAN (U-Net + single-scale D)	CelebA-HQ
2	Baseline + Mask R-CNN +perp loss+ Pixel wise loss+ multiscale discriminator	CelebA-HQ
3	Experiment 2 +Attention	CelebA-HQ
4	Experiment 3+FM loss+ SSIM loss(**proposed**)	CelebA-HQ

Table 5.6 define the different experimental categories for testing the effectiveness of various architectural improvements in the proposed face colorization architecture. The same CelebA-HQ dataset is used in all the experiments for maintaining data uniformity in terms of quality and

diversity. A U-Net generator based basic GAN framework with a single-scale discriminator is used as the baseline model. In experiment 2, a facial semantic mask generated by a pretrained Mask R-CNN is used as additional input to guide the generator. In addition, perceptual loss and pixel-wise loss are incorporated into the objective function for enhancing visual quality and authenticity. A multi-scale discriminator is also introduced to identify both local and global information. Experiment 3 is a refinement of the 2nd Experiment with additional incorporation of attention mechanisms, such as channel attention in the encoder and self-attention in the decoder, to enable the generator to effectively learn the local and global features of the face. Last experiment proposes a model that combines all the above experiment's enhancements and adds FM loss to promote convergence and realism, and SSIM loss to maintain perceptual structural similarity between actual and output images. This combined setting attempts to maximize resulting color facial image fidelity and reality using both structure knowledge and consciousness of context. Through conducting these controlled experiments, one can systematically examine the contribution of each component to the success of the entire generation process

CHAPTER SIX

6. RESULTS AND DISCUSSION

6.1 Overview of the Chapter

This chapter exhibits the outcome of the baseline study and experiments that were conducted, together with our method. The outcome is presented in the form of graphs, charts, and tables to make easy comparisons and understandings.

6.2 Experimental Results

The model was trained on various iterations with certain parameters varied between tests. Four experiments were conducted on the same hardware setup, software platform, and dataset to yield a similar and reliable result. The model was compared on the CelebA-HQ dataset, where the proposed model achieved higher quantitative performance according to standard evaluation metrics. Further, it turned out to be able to generate highly realistic facial details such as hair, skin, and eyes, as testified by comparison to the best current methods.

6.3 Results Based on Evaluation Metrics

For the assessment of the model in an objective manner, we employed three metrics: Structural Similarity Index (SSIM), Peak Signal-to-Noise Ratio (PSNR), and Fréchet Inception Distance (FID). As indicated by (Tu et al., 2022), we employed Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) to quantify our model. PSNR estimates the quality of the created face view in preserving the overall structure and face information of the original facial image. And, by way of more than mere measurement of noise levels, as SSIM estimates the extent to which the synthesized face view can estimate the spatial detail of the original face, i.e., where the eyes are, the nose, and the mouth shape. FID, on the other hand, estimates perceptual realism and distribution similarity between the real and the synthesized images. Therefore, we believe that these measures are effective in comparing the performance of the proposed model. PSNR and SSIM, and FID score of experiments performed on the CelebA-HQ dataset are presented in Tables 6.1 and 6.2, respectively. The greater PSNR, SSIM, and the smaller FID, the more generated faces approach the originals in terms of quality and consistency. More PSNR and SSIM values, as well as fewer FID score values, mean that the synthesized faces are very close to the original faces in

both quality and consistency.

6.3.1 Evaluation of Experiments on the CelebA-HQ dataset

To identify the best result in the experiment, we've highlighted the best results in bold

Table 6.1: PSNR and SSIM evaluation on the CelebA-HQ dataset

S. No	***Model***	***SSMI***	***PSNR***
1	Baseline GAN (U-Net + single-scale D)	0.8523	20.2462
2	Baseline + Mask R-CNN +perp loss+ Pixel wise loss+ multiscale discriminator	0.8688	26.8985
3	Experiment 2 +self-attention +channel attention	0.8813	26.1235
4	Experiment 3+FM loss+ SSIM loss(**proposed**)	**0.9207**	**28.3258**

Therefore, as provided in Table 6.1, our proposed model (experiment-4) approach outperforms all state-of-the-art approaches with the CelebA-HQ dataset image by improving PSNR values from 20.2462 to 28.3258 and SSIM values from 0.8523 to 0.9207.

Table 6. 2: FID evaluation on CelebA-HQ dataset

S. No	***Model***	***FID***
1	Baseline GAN (U-Net + single-scale D)	36.2316
2	Baseline + Mask R-CNN +perp loss+ Pixel wise loss+ multiscale discriminator	31.0512
3	Experiment 2 +self-attention +channel attention	24.2312
4	Experiment 3+fm loss+ SSIM loss(**proposed**)	**21.7712**

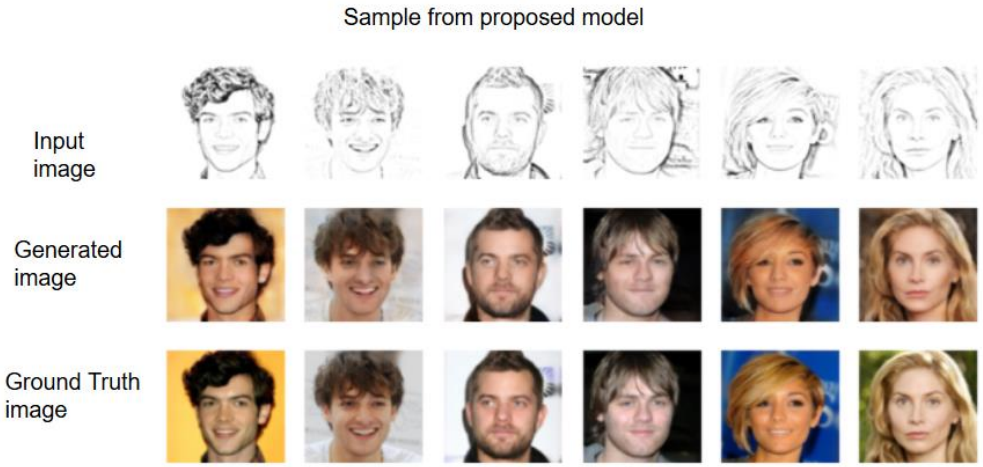
To identify the best result in the experiment, we've identified the best results in bold.

To determine the best configuration, we shaded the best-performing model with the smallest FID value. From Table 6.2, we can observe that our proposed model (Experiment 4), involving a U-Net generator, attention mechanism, multi-scale discriminator, and Mask R-CNN, achieved maximum performance with an FID score of 21.7712. This indicates a significant improvement in both perceptual quality and realism of the generated images compared to other experimental configurations.

6.4 Generated Image Results of Experiments

Here we present the output produced facial images from each experiment, including the baseline model. We have presented in the first row the input grayscale facial images, followed by the subsequent rows with the colorized facial outputs in each experimental setting. We have presented in the bottom-most row the associated ground truth color facial images, allowing visual comparison of the produced outputs with the original images.

6.4.1 Generated Results of Experiments for the CelebA-HQ dataset



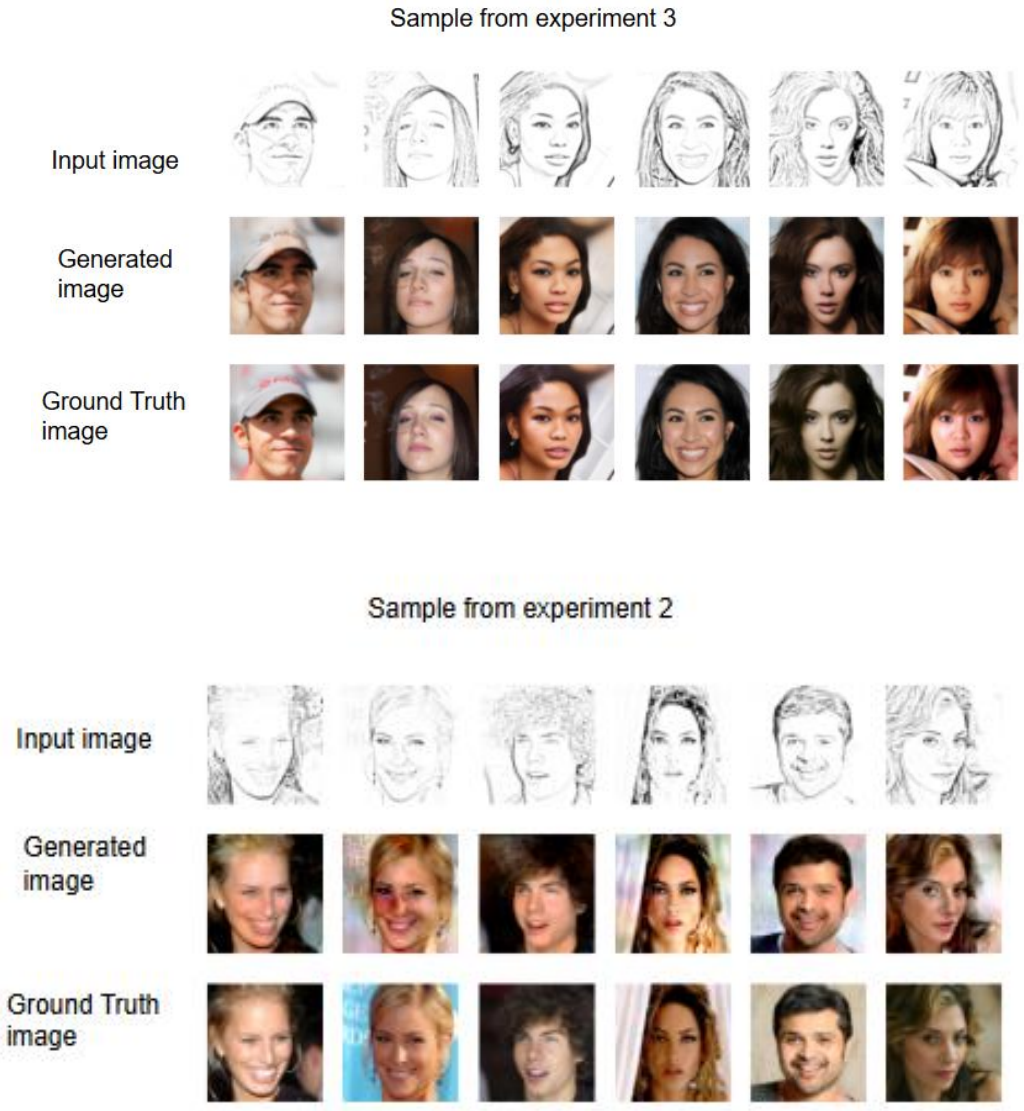


Figure 6. 1: Generated facial images in different experiment classes

6.5 Sample Training Log for Color Face Generation Using Mask R-CNN-GAN

To monitor the performance of the proposed Mask R-CNN-GAN model while it was being trained, we used graphs and charts of key parameters in learning. Training log values monitored parameters such as generator loss and discriminator loss, and reconstruction and perceptual losses where necessary. Monitoring these values provided us with insightful feedback on how much the model was minimizing errors with time. Aside from that, the logs were informative about the stability of training, the behavior of convergence, and the performance of the model in general in generating

high-quality color facial images.

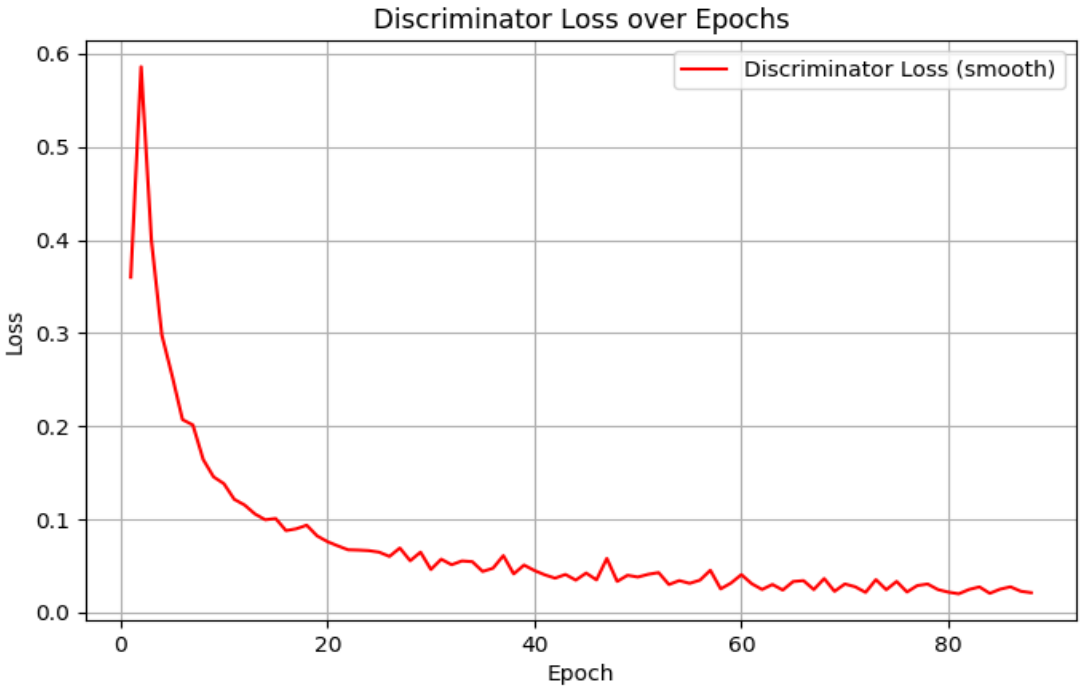


Figure 6. 2: Discriminator loss during training on the CelebA-HQ dataset

Figure 6.2 illustrates Discriminator Loss over epochs during training using the CelebA-HQ dataset. Loss is very high in the beginning, representing the high ability of the discriminator to differentiate between the real and fake image, particularly when the generator output is of low quality. In the later epochs, there is a huge drop in loss, reflecting progress in the generator learning to generate more realistic images that are hard for the discriminator to distinguish. Following the drop, the loss oscillates within a narrow range with small increases and drops typical of adversarial training dynamics, with both networks constantly changing. General stability indicates that the training is nicely balanced and that one network is not overpowering the other.

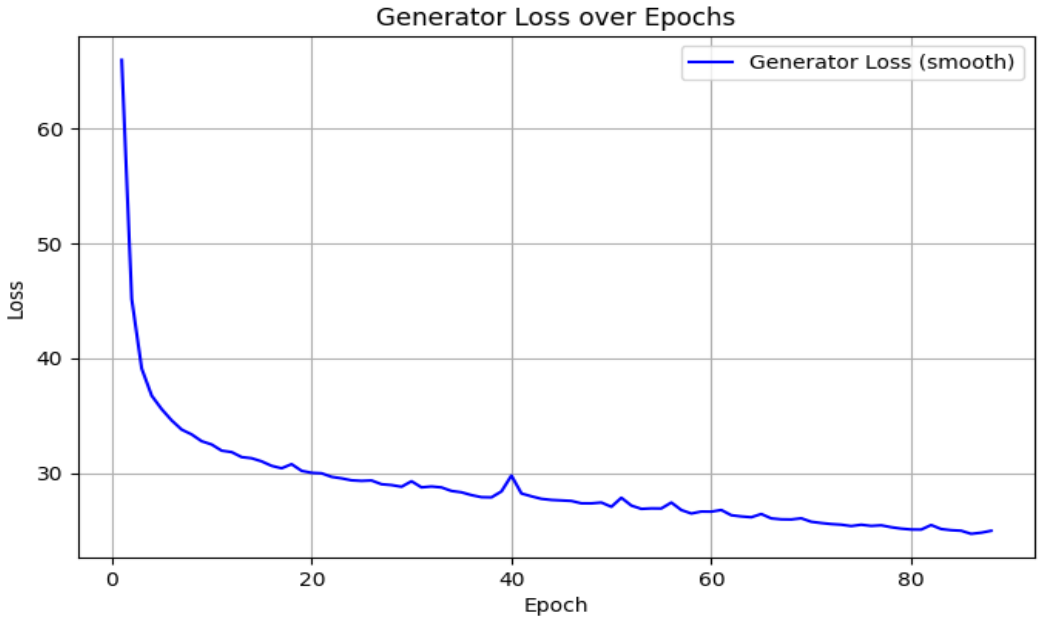


Figure 6.3: Generator loss during training on the CelebA-HQ dataset

Figure 6.3 depicts the generator Loss over epochs for the same training iteration. The loss is high at the start, which indicates that the generator is unable to generate realistic images during the initial phase. However, the loss decreases gradually as it becomes trained, with the highest drop felt within the first few epochs. This downward trend indicates the steady improvement of the generator in generating outputs increasingly misleading the discriminator. The gradual fall, not too wildly fluctuating, is an indication of consistent learning and effective optimization.

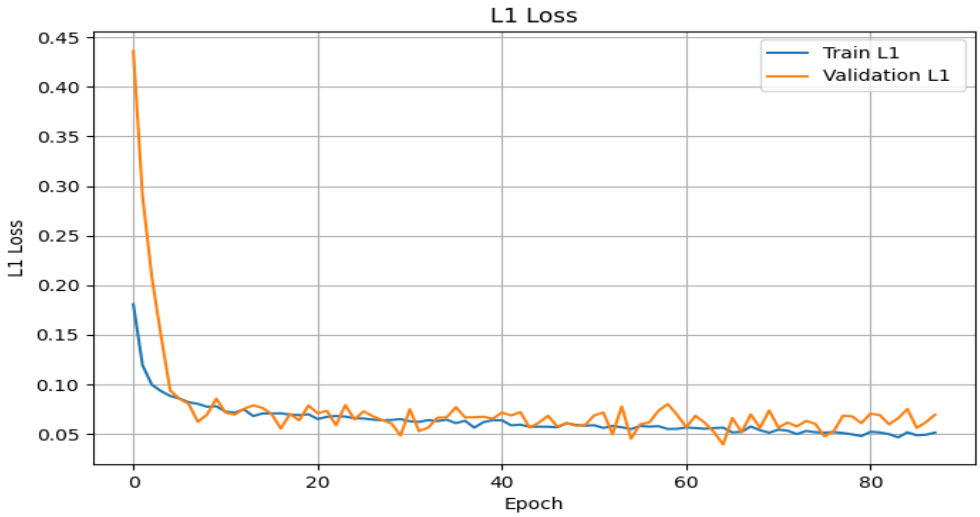


Figure 6.4: L1 loss during training and validation on the CelebA-HQ dataset

As Figure 6.4 shows, the L1 loss decreases considerably, which means the generator learns pixel-wise minimization of error between the generated and real images. During later training, the loss decreases further but at a diminishing rate, which indicates further improvement in structure alignment. The training as well as the validation losses fluctuate over epochs, which reflects pixel-level variability as well as discrepancy in the consistency of reconstructions, but the overarching diminishing trend suggests steady learning.

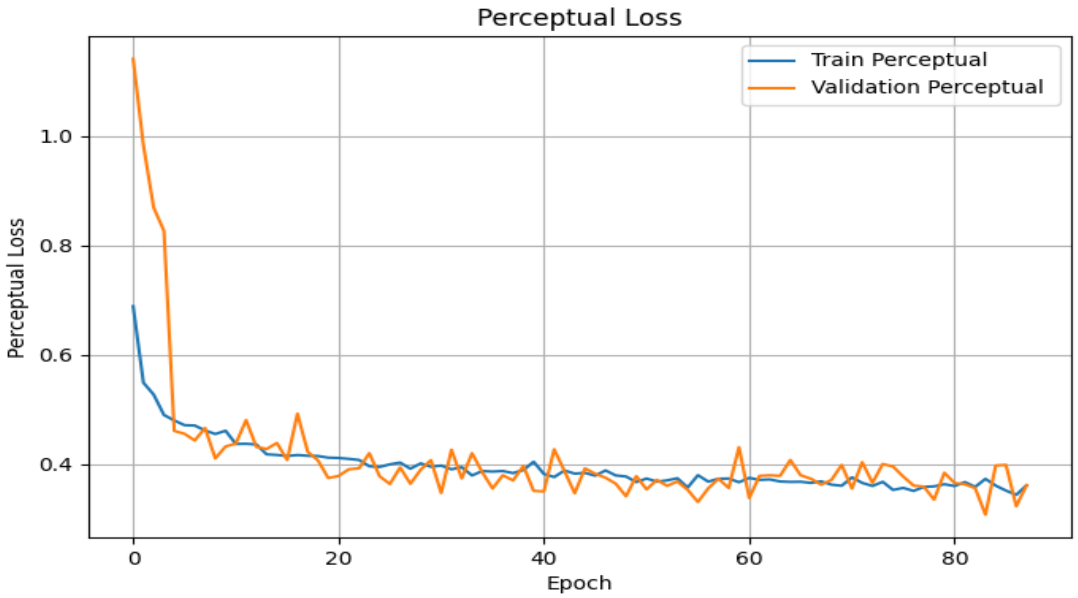


Figure 6. 5: Perceptual loss during training and validation on the CelebA-HQ dataset

Figure 6.5 illustrates the Perceptual Loss over epochs during training on the CelebA-HQ dataset. In the initial epochs, the loss drops sharply, indicating that the generator is quickly learning to produce images with perceptual features closer to those of real images. After this rapid early improvement, the loss continues to decline more gradually, reflecting ongoing refinements in high-level visual quality. Minor fluctuations appear in later epochs, but the overall trend remains downward, suggesting stable learning and improved perceptual similarity between generated and real samples over time.

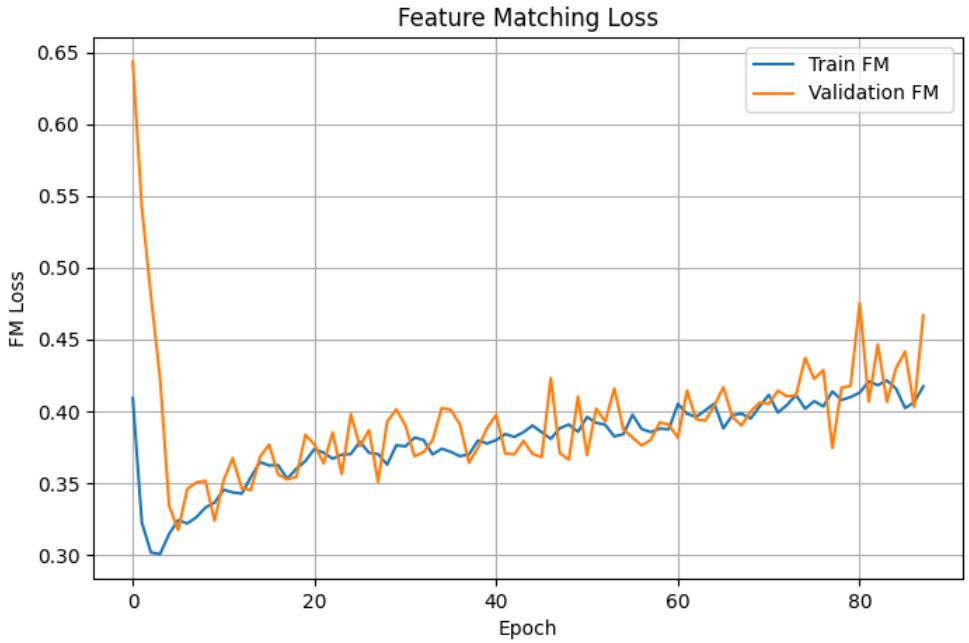


Figure 6. 6 FM Loss during training and validation on the CelebA-HQ dataset

Figure 6.6 shows the trend for FM Loss during training on the CelebA-HQ dataset. The FM loss is high at the start of training, reflecting a large difference between the internal feature representations of real and synthesized images as discerned by the discriminator. The loss decreases steadily as training continues, demonstrating that the generator is learning to generate outputs whose feature representations become more similar to those of real images. The smooth decreasing trend indicates stable training and reflects both global structure and fine-grained texture consistency improvements in the generated outputs.

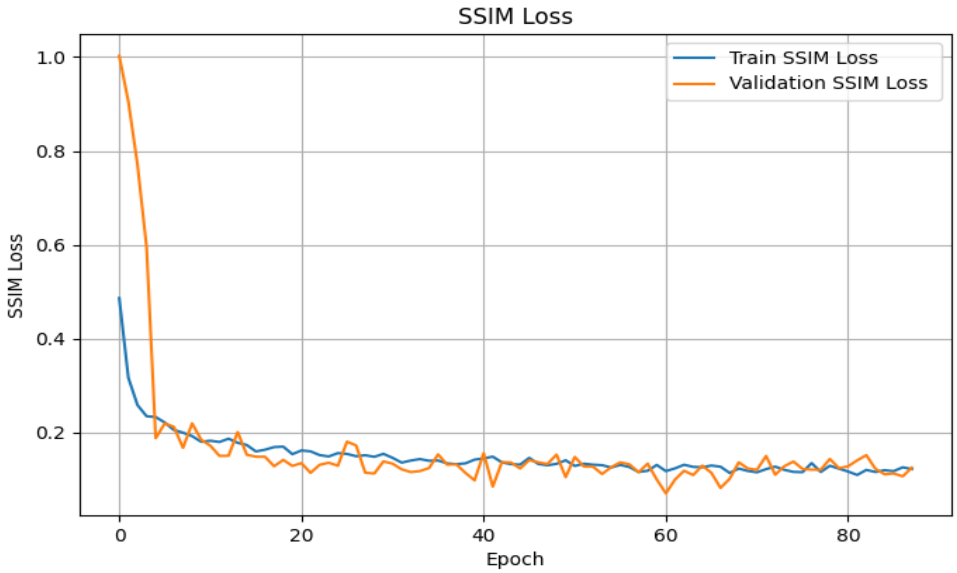


Figure 6. 7 SSIM Loss during training and validation on the CelebA-HQ dataset

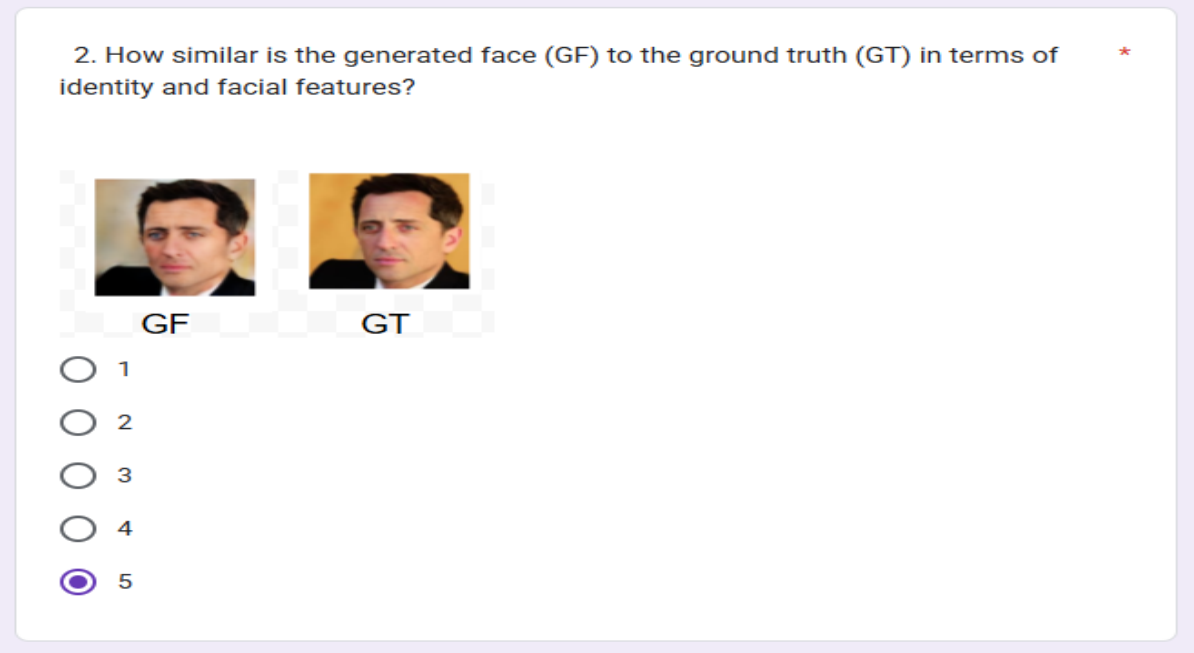
Figure 6.7 presents the SSIM loss curve over training epochs on the CelebA-HQ dataset. The curve rises steeply in the initial few epochs, which implies that the model learns rapidly to preserve structural information in the outputs. With prolonged training, the SSIM measure keeps rising steadily, which shows high structural fidelity among the outputs and ground truth images. The smooth positive trend with minimal fluctuations indicates consistent training dynamics, and the saturation observed after about 90 epochs signifies that the model has converged well.

6.6 Human Perceptual Evaluation

Human perception evaluation was performed to evaluate the quality of face images (GF) synthesized by the novel hybrid deep learning architecture that combines Mask R-CNN and GAN. Such a subjective assessment accompanies quantitative measurements like FID, SSIM, and PSNR to capture image quality's qualitative aspects that involve realism, preservation of identity, sharpness, and clearness of features like skin, eyes, and hair, resemblance to a natural human face, and overall quality, considering all these aspects. Five participants were asked to evaluate the synthesized face images alongside ground truth (GT) faces and rated each image on a subjective score ranging between 1 and 5 (1 = Poor, 5 = Excellent) on five distinct aspects: realism and naturalistic appearance of synthesized face, resemblance to GT in terms of identity and facial features, sharpness and clearness of features like skin, eyes, and hair, resemblance to a natural human face, and overall quality considering all these aspects. All participants' ratings were pooled

and averaged against each criterion, producing a numerical human perception evaluation and unveiling insights about how successful the developed model is in synthesizing visually realistic and identity-preserving faces.

As an illustration, one of the evaluation questions presented to participants through Google Forms is shown below:



The response of the participant is collected through a Google Form as follows:

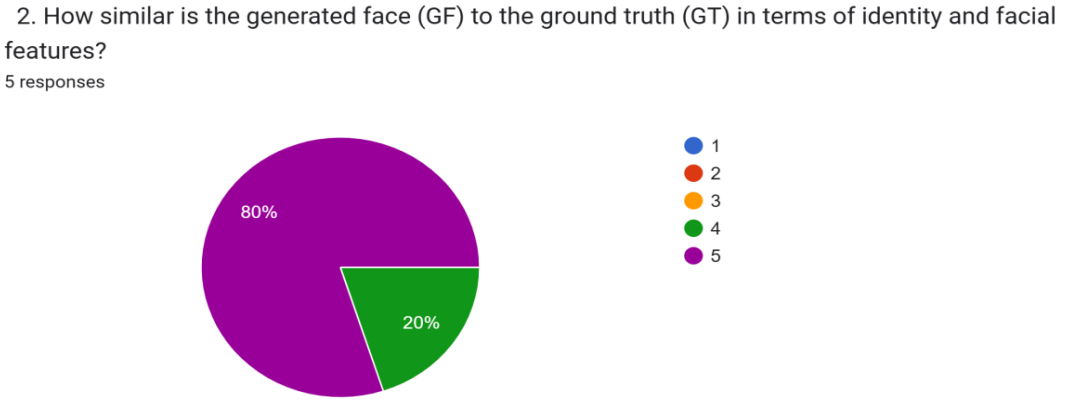


Figure 6.8 Sample of Responses from participants

As shown in Figure 6.8, 80% of participants rated the generated face as "Excellent (5)," while 20% rated it as "Good (4)," resulting in an average score of 4.8 out of 5 for this criterion.

The responses from all participants were pooled and averaged across the five evaluation aspects.

The summarized results are presented in Table 6.3.

Table 6.3 Human Perceptual Evaluation Scores from Five Participants

Evaluation Criteria	Participant one	Participant two	Participant three	Participant four	Participant five	Average
Realism	5	5	4	5	5	5
Similarity to GT	5	4	5	5	5	4.8
Clarity	4	5	5	5	5	4.8
Resemblance to the real face	5	5	5	5	4	4.8
Overall Quality	4	5	5	5	5	4.8

Table 6.3 presents the human perceptual rating scores of the synthesized face images (GF) by five subjects. The subjects rated the images using a 1–5 scale (1 = Poor, 5 = Excellent) against five criteria: realism and natural appearance, similarity to the ground truth (GT), sharpness and acuteness of details, similarity to an actual human face, and overall quality. The mean scores indicate that the generated images were all considered to be realistic, clear, and highly similar to real human faces, clearly demonstrating the effectiveness of the proposed hybrid Mask R-CNN and GAN model.

6.7 Result Discussion

Experimental results verify that the proposed hybrid Mask R-CNN and GAN approach can successfully synthesize realistic color faces at high resolution. Quantitative evaluation by PSNR, SSIM, and FID is enhanced in all experiments. A baseline GAN (U-Net + single-scale discriminator) had PSNR = 20.2462 and SSIM = 0.8523. Adding Mask R-CNN, pixel-wise loss, and perceptual loss functions, the multi-scale discriminator (Experiment 2) led to PSNR 26.8985 and SSIM 0.8688. Adding channel attention mechanism as well as self-attention (Experiment 3) bettered the SSIM to 0.8813. Lastly, the top-proposed model (Experiment 4) with all the optimizations, feature-matching and SSIM loss, recorded PSNR 28.3258, SSIM 0.9207, and FID

21.7712, which is much better than the baseline and the other experiments. This clearly indicates the effectiveness of semantic guidance, attention, and multi-scale discrimination to enhance structural correctness, color accuracy, and general realism of the faces being generated.

The FID scores also validate, indicating the distribution of synthesized images and actual facial images intersect earlier than in any other setup. The drop from 36.2316 (baseline) to 21.7712 (proposed model) reflects greater perceptual realism. All architectural improvements stepwise boost generator capability to generate visually realistic and structurally sound faces.

Human perceptual testing also confirms the strength of the proposed model. Five users scored synthesized faces on five aspects: realism, similarity to ground truth, clearness of features, closeness to real human faces, and quality. The mean score was 4.8/5, and 80% of users scored realism as "Excellent." The results affirm that generated faces are visually accurate, maintain one's identity, and adequately depict facial features like eyes, mouth, hair, and skin.

Overall, human judgment and objective evaluation both indicate the hybrid Mask R-CNN and GAN approach outperforms baseline GANs and addresses typical face generation issues like structural inconsistency, blurry details, and over-realistic colorization. Mask R-CNN gives precise semantic guidance, attention mechanisms remove long-range dependencies, and a multi-scale discriminator produces realistic textures at all resolutions. Their synergistic union renders the model presented here a feasible solution to realistic and high-fidelity facial coloring and one that can be extended to facilitating forensic examination, historical photo reconstruction, and other visually realistic and identity-appropriate facial image applications.

6.8 Research Question and Answer Discussion

Solution to Q1. The impact of integrating Mask R-CNN on the quality of color face generation is investigated in this research. By incorporating Mask R-CNN as a semantic feature extractor, the synthesized color face image quality is significantly improved. By providing accurate segmentation masks of structurally salient facial regions such as the eyes, nose, mouth, and jawline, Mask R-CNN instructs the generator to focus on structurally and semantically significant regions. This results in improved structural accuracy, better feature preservation, and more realistic coloring, as demonstrated by higher PSNR and SSIM scores and reduced FID scores.

Solution to Q2: Improvements in different methods were applied within the GAN model to improve the quality of the generated faces. Decoder self-attention preserves long-range

dependencies and fine facial details, and encoder channel attention is centered on the most informative channels of features. Multi-scale PatchGAN discriminators enforce reality on multiple scales, and pixel-wise, perceptual, feature-matching, and SSIM losses together guide the training process to visual quality and structural coherence improvement of the synthetic images.

Solution to Q3. Realism and image quality are improved with the hybrid compared to single GANs. Through the exploitation of the strengths of modules such as Mask R-CNN, attention, and multi-scale discriminators, the hybrid model can better localize facial features, texture data, and structural consistency. Quantitatively, it also yields better PSNR and SSIM values and significantly lower FID values, which mark improved fidelity and perception. By naked eye, the generated faces are more realistic with fewer artifacts and better color representation of the most significant facial regions like eyes, lips, and skin color. Overall, the hybrid method produces a better and more stable solution for colored face generation of high quality.

CHAPTER SEVEN

7. CONCLUSION AND FUTURE WORKS

7.1 Conclusion

Face image synthesis in high-resolution, true-world colors is a difficult computer vision task wherein both color accuracy and structural consistency need to be preserved. Existing GAN-based methods, excellent as they are for generating beautiful images, do not mimic finer facial structures such as eye shape, skin texture, hair structure, and more subtle features typical of the natural human face. These constraints are most likely to result in artifacts, soft regions, and non-uniform color distribution, reducing the global realism of output images. The hybrid deep learning architecture proposed combined a U-Net GAN with semantics-guided pre-trained Mask R-CNN as well as attention mechanisms. Semantic masks that marked salient facial regions like eyes, nose, mouth, hair, and skin were combined with grayscale input sketches to produce semantically informative inputs to guide the generator to generate structurally proper and visually coherent face images. Channel attention in the encoder concentrates information channels of features, and self-attention in the decoder chases long-distance dependencies to maintain global facial coherence.

The model was extensively tested on the CelebA-HQ dataset in experiments. PSNR, SSIM, and FID quantitative metrics were attained because the hybrid model outperformed baseline GANs by an enormous margin of 28.3258, SSIM 0.9207, and FID 21.7712, indicating enhanced image quality and structure similarity. Apart from the above objective measures, perception experiments were also performed using human subjects, where synthesized faces were checked for realism, color accuracy, and coherence of structure by the observers. Experiments confirmed that images produced by the hybrid model appeared more natural and pleasing than baseline and state-of-the-art generation methods. Visual inspection revealed that the model had restored hard facial features like eyes, mouth, hair, and skin with less artifacts and more consistent color distribution.

It does this with semantic guidance, attention, multi-scale PatchGAN discrimination, and sophisticated loss functions like pixel-wise, perceptual, feature-matching, and SSIM. The inclusion of the inclusion also comes with local texture preservation and global structure

preservation, allowing realistic facial images to be generated for real-world applications. Beyond quantitative performance, the article brings to light human visual judgment as an important supplement to objective quantifications in forensic examination, judicial examination, and historical image reconstruction, where human visual judgment in the spirit of realism is most urgently called for. Remnants of failure continue to exist.

The model was only trained on the CelebA-HQ dataset, which contained largely celebrity faces and is restricted to generalizability to other groups. The model is yet to be optimized for real-time video synthesis or dynamic facial applications. High computational expense, a side effect of attention mechanisms and multi-scale discrimination, also restricts use and accessibility in scope. But the paper confirms that semantic segmentation, attention modules, and multiscale adversarial learning actually play a significant role in the enhancement of realism, structure consistency, and perceptual quality of color face images, and are a good foundation for subsequent and real applications in the future.

7.2 Future Works

Future research could be directed towards further enhancing the proposed model in several ways: further enhancing segmentation accuracy, consolidating multiple-modal datasets, enabling real-time deployment, generating ultra-high-resolution images, and applying user-controllable variations to enhance the proposed framework further. Future research can be as follows:

- Improved Segmentation Accuracy: Applying the state-of-the-art segmentation models, i.e., Mask2Former or SAM, will produce much better and transferable semantic masks.
- Multi-modal input: Extending the input of the GAN to something other than the other modality, i.e., depth maps or pose estimations, so that it would be invariant to different conditions and would augment or enhance its robustness across multiple factors.
- Real-time and edge models: Optimization of the model so that it may be deployed on the edge or do real-time computation through pruning, quantization, and knowledge distillation.
- Ultra-high-resolution synthesis: Adopting advance or diffusion-based generation techniques for creating ultra-high-resolution face images without sacrificing realism.
- Diversity control: Applying style-based control or latent space manipulation techniques for user-controlled diversity in the synthesis image.

Lastly, it's proven here in this paper that combined Mask R-CNN segmentation ability and GAN generation ability is definitely an amazing way for realistic and structured face image production. The result has vast potential to find its application in digital avatars, face reconstruction, virtual reality, and multimedia content generation, as well as an effective introduction for further research in controllable and effective image creation systems.

SPECIAL ACKNOWLEDGMENT

This research work was funded by a grant number ASTU/SM-R/1121/25 from Adama Science and Technology University.

REFERENCE

Alzubaidi, Laith, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, et al. 2021. "Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions." *Journal of Big Data* 8: 1–74.

Arjovsky, Martin, Soumith Chintala, and Léon Bottou. 2017. "Wasserstein GAN." http://arxiv.org/abs/1701.07875.

Bharati, Puja, and Ankita Pramanik. 2020. "Deep Learning Techniques—R-CNN to Mask R-CNN: A Survey BT - Computational Intelligence in Pattern Recognition." In eds. Asit Kumar Das, Janmenjoy Nayak, Bighnaraj Naik, Soumen Kumar Pati, and Danilo Pelusi. Singapore: Springer Singapore, 657–68.

Bolya, Daniel, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. 2019. "YOLACT: Real-Time Instance Segmentation." *Proceedings of the IEEE International Conference on Computer Vision*: 9156–65. doi:10.1109/ICCV.2019.00925.

Chang, Yeong-Hwa, Pei-Hua Chung, Yu-Hsiang Chai, and Hung-Wei Lin. 2024. "Color Face Image Generation with Improved Generative Adversarial Networks." *Electronics* 13(7). doi:10.3390/electronics13071205.

Combinations, Space Channel. 2022. "Space Channel Combinations." : 1–13.

Daras, Giannis, Augustus Odena, Han Zhang, and Alexandros G Dimakis. 2020. "Your Local GAN: Designing Two-Dimensional Local Attention Mechanisms for Generative Models." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),*.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. "Generative Adversarial Nets." In *Advances in Neural Information Processing Systems*, eds. Z Ghahramani, M Welling, C Cortes, N Lawrence, and K Q Weinberger. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Grnarova, Paulina, Kfir Y. Levy, Aurelien Lucchi, Nathanaël Perraudin, Ian Goodfellow, Thomas Hofmann, and Andreas Krause. 2019. "A Domain Agnostic Measure for Monitoring and Evaluating GANs." *Advances in Neural Information Processing Systems* 32(NeurIPS).

Guo, Yong, Qi Chen, Jian Chen, Qingyao Wu, Qinfeng Shi, and Mingkui Tan. 2019. "Auto-Embedding Generative Adversarial Networks for High Resolution Image Synthesis." *IEEE Transactions on Multimedia* 21(11): 2726–37.

He, Kaiming, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. 2017. "Mask R-CNN." In *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, 2961–69.

Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler und Sepp Hochreiter. 2017. "Gans Trained by a Two-Time-Scale Update Rule Converge to a Local Nash Equilibrium." *Advances in Neural Information Processing Systems* 30.

Hsu, Chih-Chung, Yi-Xiu Zhuang, and Chia-Yen Lee. 2020. "Deep Fake Image Detection Based on Pairwise Learning." *Applied Sciences* 10(1). doi:10.3390/app10010370.

Hu, Jie, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. 2020. "Squeeze-and-Excitation Networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42(8): 2011–23. doi:10.1109/TPAMI.2019.2913372.

Isola, Phillip, Jun Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2017. "Image-to-Image Translation with Conditional Adversarial Networks." *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* 2017-Janua: 5967–76. doi:10.1109/CVPR.2017.632.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei. 2016. "Perceptual Losses for Real-Time Style Transfer and Super-Resolution." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9906 LNCS: 694–711. doi:10.1007/978-3-319-46475-6_43.

Karras, Tero, Samuli Laine, and Timo Aila. 2019. "A Style-Based Generator Architecture for Generative Adversarial Networks." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4401–10.

Karras, Tero, Samuli Laine, and Timo Aila. 2021. "A Style-Based Generator Architecture for Generative Adversarial Networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43(12): 4217–28. doi:10.1109/TPAMI.2020.2970919.

Karras, Tero, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. "Analyzing and Improving the Image Quality of Stylegan." In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8110–19.

Kaushik, C P, and Shailendra Singh. 2022. "3. Generate Artificial Human Faces with Deep

Convolutional Generative Adversarial Network (DCGAN) Machine Learning Model." doi:10.1007/978-981-99-5974-7_5.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. "ImageNet Classification with Deep Convolutional Neural Networks." In *Advances in Neural Information Processing Systems*, eds. F Pereira, C J Burges, L Bottou, and K Q Weinberger. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

LeCun, Y, B Boser, J S Denker, D Henderson, R E Howard, W Hubbard, and L D Jackel. 1989. "Backpropagation Applied to Handwritten Zip Code Recognition." *Neural Computation* 1(4): 541–51. doi:10.1162/neco.1989.1.4.541.

Li, Haodong, Bin Li, Shunquan Tan, and Jiwu Huang. 2020. "Identification of Deep Network Generated Images Using Disparities in Color Components." *Signal Processing* 174: 107616. doi:https://doi.org/10.1016/j.sigpro.2020.107616.

Lin, Tsung Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. "Microsoft COCO: Common Objects in Context." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8693 LNCS(PART 5): 740–55. doi:10.1007/978-3-319-10602-1_48.

Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. "Deep Learning Face Attributes in the Wild." In *Proceedings of the IEEE International Conference on Computer Vision (ICCV),*.

Mao, Xudong, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. 2017. "Least Squares Generative Adversarial Networks." *Proceedings of the IEEE International Conference on Computer Vision* 2017-Octob: 2813–21. doi:10.1109/ICCV.2017.304.

Marra, Francesco, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. 2018. "Detection of GAN-Generated Fake Images over Social Networks." *Proceedings - IEEE 1st Conference on Multimedia Information Processing and Retrieval, MIPR 2018*: 384–89. doi:10.1109/MIPR.2018.00084.

Miranda-Hernández, J, M Castelán, and L A Torres-Méndez. 2012. "Face Colour Synthesis Using Partial Least Squares and the Luminance-$α$-$β$ Colour Transform." *IET Computer*

Vision 6(4): 263–72.

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. "Spectral Normalization for Generative Adversarial Networks." *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*.

Radford, Alec. 2015. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks." *arXiv preprint arXiv:1511.06434*.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. 2015. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In *Medical Image Computing and Computer-Assisted Intervention--MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, 234–41.

Rosenfeld, Azriel. 1988. "Computer Vision: Basic Principles." *Proceedings of the IEEE* 76(8): 863–68.

S Karthika, and M Durgadevi. 2021. "Generative Adversarial Network (GAN): A General Review on Different Variants of GAN and Applications." In *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, 1–8. doi:10.1109/ICCES51350.2021.9489160.

Salimans, Tim, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. 2016. "Improved Techniques for Training GANs." In *Advances in Neural Information Processing Systems*, eds. D Lee, M Sugiyama, U Luxburg, I Guyon, and R Garnett. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.

Samuel, A L. 1959. "Some Studies in Machine Learning Using the Game of Checkers." *IBM Journal of Research and Development* 3(3): 210–29. doi:10.1147/rd. 33.0210.

Shu, Jian-Hua, Fu-Dong Nian, Ming-Hui Yu, and Xu Li. 2020. "An Improved Mask R-CNN Model for Multiorgan Segmentation." *Mathematical Problems in Engineering* 2020(1): 8351725.

Simonyan, Karen, and Andrew Zisserman. 2015. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*: 1–14.

Sun, Jianyuan, Hongchuan Yu, Jian J Zhang, Junyu Dong, Hui Yu, and Guoqiang Zhong. 2022.

"Face Image-Sketch Synthesis via Generative Adversarial Fusion." *Neural Networks* 154: 179–89.

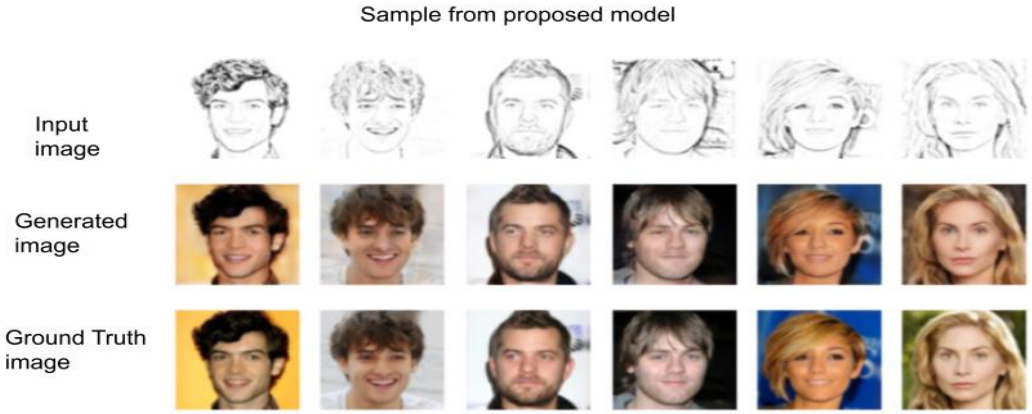
Wang, Lidan, Vishwanath Sindagi, and Vishal Patel. 2018. "High-Quality Facial Photo-Sketch Synthesis Using Multi-Adversarial Networks." In *2018 13th IEEE International Conference on Automatic Face \& Gesture Recognition (FG 2018),* 83–90.

Woo, Sanghyun, Jongchan Park, Joon Young Lee, and In So Kweon. 2018. "CBAM: Convolutional Block Attention Module." *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11211 LNCS: 3–19. doi:10.1007/978-3-030-01234-2_1.

Zhang, Han, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. 2019. "Self-Attention Generative Adversarial Networks." *36th International Conference on Machine Learning, ICML 2019* 2019-June: 12744–53.

Appendix

Appendix A: CelebA-HQ Test Result

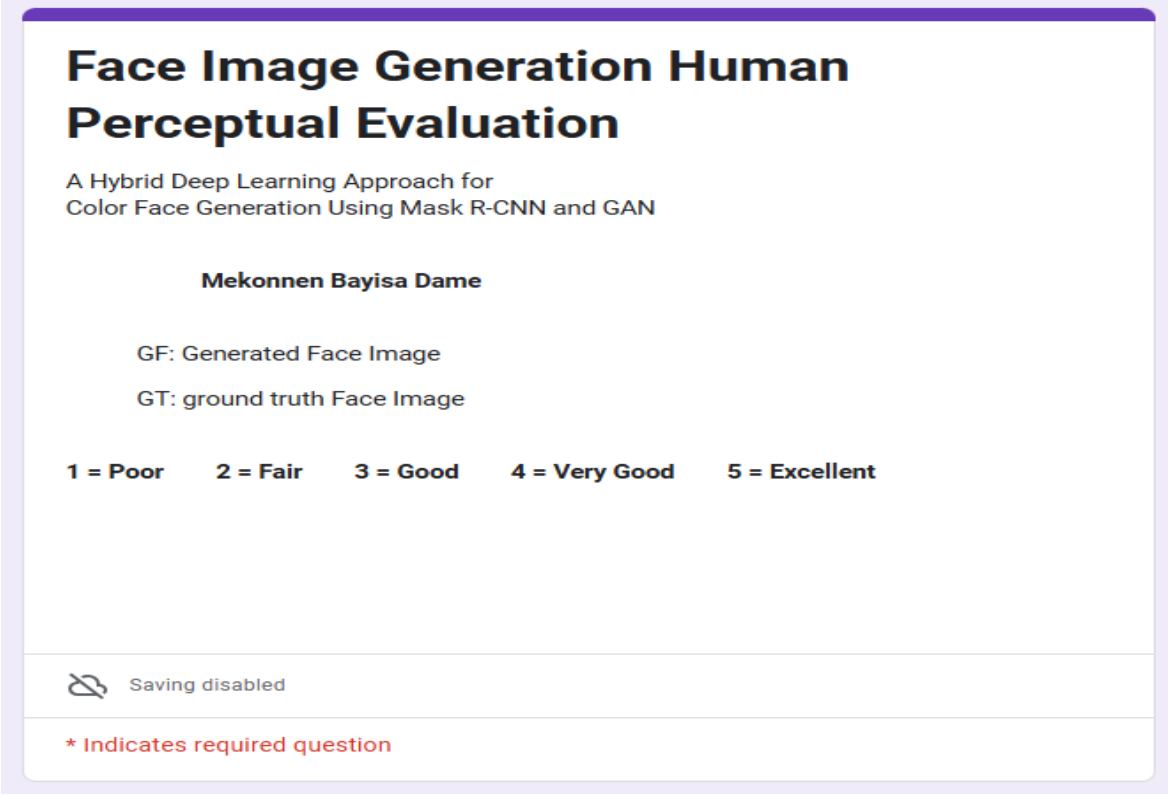


Appendix B: Sample Code

```
# Generator: Attention-enhanced U-Net
class UNetAttentionEnhanced(nn.Module):
    def __init__(self, in_channels=2, out_channels=3, base=64):
        super().__init__()

        # Encoder
        self.enc1 = self._enc_block(in_channels, base, use_channel_attn=True)       # 2 -> 64
        self.enc2 = self._enc_block(base, base*2, use_channel_attn=True)           # 64 -> 128
        self.enc3 = self._enc_block(base*2, base*4, use_channel_attn=True)         # 128 -> 256
        self.enc4 = self._enc_block(base*4, base*8, use_channel_attn=True)         # 256 -> 512

        # Bottleneck
        self.bottleneck = nn.Sequential(
            nn.Conv2d(base*8, base*8, 3, padding=1),
            nn.InstanceNorm2d(base*8),
            nn.ReLU(inplace=True)
        )

        # Decoder (corrected input channels)
        self.up4 = nn.ConvTranspose2d(base*8, base*4, 2, stride=2)                 # 512 -> 256
        self.dec4 = self._dec_block(base*4 + base*8, base*4, use_self_attn=True)  # 256 + 512 = 7

        self.up3 = nn.ConvTranspose2d(base*4, base*2, 2, stride=2)                 # 256 -> 128
        self.dec3 = self._dec_block(base*2 + base*4, base*2, use_self_attn=True)  # 128 + 256 = 3

        self.up2 = nn.ConvTranspose2d(base*2, base, 2, stride=2)                   # 128 -> 64
        self.dec2 = self._dec_block(base + base*2, base, use_self_attn=False)      # 64 + 128 = 1

        self.up1 = nn.ConvTranspose2d(base, base//2, 2, stride=2)                  # 64 -> 32
        self.dec1 = self._dec_block(base//2 + base, base//2, use_self_attn=False)  # 32 + 64 = 96

        self.final = nn.Sequential(
            nn.Conv2d(base//2, out_channels, 3, padding=1),
            nn.Tanh()
        )

    def _enc_block(self, in_ch, out_ch, use_channel_attn=False):
        layers = [
            nn.Conv2d(in_ch, out_ch, 4, stride=2, padding=1),
            nn.InstanceNorm2d(out_ch),
            nn.LeakyReLU(0.2, inplace=True),
            ResidualBlock(out_ch)
        ]
        if use_channel_attn:
            layers.append(ChannelAttention(out_ch))
        return nn.Sequential(*layers)
```

```
def _dec_block(self, in_ch, out_ch, use_self_attn=False):
    layers = [
        nn.Conv2d(in_ch, out_ch, 3, padding=1),
        nn.InstanceNorm2d(out_ch),
        nn.ReLU(inplace=True),
        ResidualBlock(out_ch)
    ]
    if use_self_attn:
        layers.append(SelfAttention(out_ch))
    return nn.Sequential(*layers)

def forward(self, x):
    # Encoder
    e1 = self.enc1(x)
    e2 = self.enc2(e1)
    e3 = self.enc3(e2)
    e4 = self.enc4(e3)

    # Bottleneck
    b = self.bottleneck(e4)

    # Decoder with upsampling + concatenation
    up4 = self.up4(b)
    e4r = F.interpolate(e4, size=up4.shape[2:], mode='bilinear', align_corners=False)
    d4 = self.dec4(torch.cat([up4, e4r], dim=1))

    up3 = self.up3(d4)
    e3r = F.interpolate(e3, size=up3.shape[2:], mode='bilinear', align_corners=False)
    d3 = self.dec3(torch.cat([up3, e3r], dim=1))

    up2 = self.up2(d3)
    e2r = F.interpolate(e2, size=up2.shape[2:], mode='bilinear', align_corners=False)
    d2 = self.dec2(torch.cat([up2, e2r], dim=1))

    up1 = self.up1(d2)
    e1r = F.interpolate(e1, size=up1.shape[2:], mode='bilinear', align_corners=False)
    d1 = self.dec1(torch.cat([up1, e1r], dim=1))

    return self.final(d1)
```

Appendix C: Sample Results of Human Perceptual Evaluation assessment using Google Form



Face Image Generation Human Perceptual Evaluation

A Hybrid Deep Learning Approach for
Color Face Generation Using Mask R-CNN and GAN

Mekonnen Bayisa Dame

GF: Generated Face Image

GT: ground truth Face Image

1 = Poor **2 = Fair** **3 = Good** **4 = Very Good** **5 = Excellent**

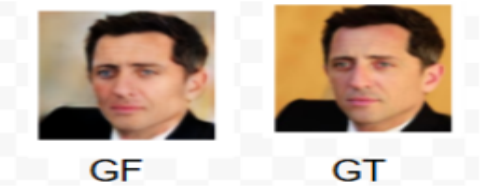
Saving disabled

* Indicates required question

Email *

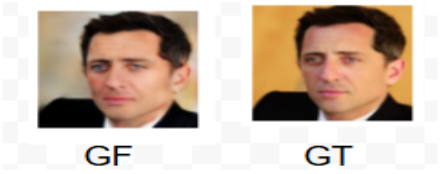
mulliemit@gmail.com

1. How realistic and natural does the generated face image (GF) appear? *



- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ◉ 5

2. How similar is the generated face (GF) to the ground truth (GT) in terms of identity and facial features? *



- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ◉ 5

3. How clear and sharp are the generated details, such as skin, eyes, and hair? *

GF

GT

- ○ 1
- ○ 2
- ○ 3
- ◉ 4
- ○ 5

4. How closely does the generated face resemble a real human face? *


GF


GT

- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ◉ 5

5. Considering all aspects (realism, similarity, clarity, and attributes), how would you rate the overall quality of the generated face (GF)? *


GF


GT

- ○ 1
- ○ 2
- ○ 3
- ◉ 4
- ○ 5

Appendix D: List of Evaluation Participants

Evaluation Participants		
S.NO	**Name**	**Field of Study**
1	Mulgeta Berhe	Computer Science and Engineering (Computer Vision)
2	Abdi Mosisa	Computer Science and Engineering (Artificial Intelligence)
3	Fatia Kedir	Computer Science and Engineering (Artificial Intelligence)
4	Kibru Alemu	Computer Science and Engineering (Data Science)
5	Tahir Aman	Computer Science and Engineering (Artificial Intelligence)

Dr. Mesfin Abebe

Thesis

Thesis proposal

Thesis proposal (MSc students)

Adama Science and Technology University

Document Details

Submission ID
trn:oid:::1:3380880337

Submission Date
Oct 21, 2025, 9:07 AM GMT

Download Date
Oct 21, 2025, 11:13 AM GMT

File Name
Final_Thesis_by_Mekonnen_Bayisa_after_modified.pdf

File Size
2.3 MB

90 Pages

20,986 Words

115,124 Characters

17% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography

Exclusions

- 1 Excluded Source

Match Groups

243 Not Cited or Quoted 14%
Matches with neither in-text citation nor quotation marks

56 Missing Quotations 3%
Matches that are still very similar to source material

1 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

11% Internet sources
14% Publications
5% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.