IMAGEGENIE: A MAGIC WAND FOR IMAGES ENHANCEMENT USING DEEP LEARNING

SUE CHEN XIANG

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

BORANG PENGESAHAN STATUS LAPORAN

JUDUL: IMAGEGENIE: A MAGIC WAND FOR IMAGES USING DEEP LEARNING

SESI PENGAJIAN: 2022 / 2023

Saya: ____SUE CHEN XIANG___________________

mengaku membenarkan tesis Projek Sarjana Muda ini disimpan di Perpustakaan Universiti Teknikal Malaysia Melaka dengan syarat-syarat kegunaan seperti berikut:

1. Tesis dan projek adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan unituk tujuan pengajian sahaja.
3. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. * Sila tandakan (✓)

___✓___ SULIT (Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

__________ TERHAD (Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi / badan di mana penyelidikan dijalankan)

__________ TIDAK TERHAD

(TANDATANGAN PELAJAR)

Alamat tetap: 157, Jln Love Lane, Kg
Baru Sg Buloh, 47000 Selangor

Tarikh: _19/9/2023_________________

(TANDATANGAN PENYELIA)

Nama Penyelia

Tarikh: ____________________________

CATATAN: * Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa.

IMAGEGENIE: A MAGIC WAND FOR IMAGES USING DEEP LEARNING

SUE CHEN XIANG

This report is submitted in partial fulfillment of the requirements for the Bachelor of Computer Science (Artificial Intelligence) with Honours.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2023

DECLARATION

I hereby declare that this project report entitled

IMAGEGENIE: A MAGIC WAND FOR IMAGES USING DEEP LEARNING

is written by me and is my own effort and that no part has been plagiarized without citations.

STUDENT : ______________________ Date : _19/9/2023___
(SUE CHEN XIANG)

I hereby declare that I have read this project report and found this project report is sufficient in term of the scope and quality for the award of Bachelor of Computer Science (Artificial Intelligence) with Honours.

SUPERVISOR : ______________________ Date : ________
(DR. FAUZIAH BINTI KASMIN)

DEDICATION

This final year project is dedicated to my supervisor, Dr. Fauziah binti Kasmin, for her guidance and support throughout this project. She has been the source of my strength and on Her wings only have I soared. I would also dedicate this to my parents who encouraged me all the way. Finally, I would like to dedicate this project to my friends for their encouragement and assistance during my university studies.

ACKNOWLEDGEMENTS

First and foremost, I would like to take this opportunity to express my gratitude to the people who helped and guided me throughout this project. I would like to show my greatest appreciation to Dr. Fauziah binti Kasmin. I can’t say thank you enough for her support and guidance. Without her encouragement and guidance this project would not have materialized.

I would like to thank my friends for their assistance, support and encouragement in completing this project. I am grateful for their constant support and help.

I would also like to thank my wonderful parents for their encouragement throughout this project. Without their motivation, I cannot fulfill this project before the deadline.

ABSTRACT

The goal of this project is to develop an application that can increase the resolution of low quality images and to perform neural style transfer that composes one image in the style of another image. In today's digital age, images play a crucial role in communication and self-expression. However, low-quality images are a common problem that many people encounter, particularly when taking pictures with their mobile phones. The lack of image quality can result in images that are blurry, pixelated, or have poor resolution. These low-quality images can be frustrating for users who want to share their images on social media platforms or use them for personal or professional purposes. In this project, several pretrained deep convolutional neural networks model that are the Enhanced Deep Residual Network (EDSR), Efficient Sub-Pixel Convolutional Neural Network (ESPCN) and deep Laplacian Pyramid Super-Resolution Network (LapSRN) are used for increasing the resolution of image and the result of each model is analyzed and compared. The EDSR was chosen as the best model as it achieved the highest average PSNR and SSIM in testing 45 images which are 25.82 dB and 0.70 respectively. Traditional image processing applications often lack the ability to perform advanced image enhancement techniques such as neural style transfer, which can be used to create artistic effects on images. The neural style transfer functionality is achieved by extracting the style of the style image using the VGG19 network architecture which is a pretrained image classification network and apply to the content image to create artistic effects on images. VGG19 was employed because it obtained the highest average ArtFID in 30 testing images which is 45.97 when compare to MobileNet and ResNet. Finally, an application is built using Flutter that combines all the functions above.

ABSTRAK

Tujuan projek ini adalah untuk membangunkan satu aplikasi yang mampu meningkatkan resolusi imej berkualiti rendah dan melakukan pemindahan gaya neural yang menggabungkan satu imej dalam gaya imej yang lain. Dalam era digital ini, imej memainkan peranan yang penting dalam komunikasi dan ekspresi diri. Walau bagaimanapun, imej berkualiti rendah merupakan masalah yang biasa dihadapi oleh ramai orang, terutamanya apabila mengambil gambar menggunakan telefon bimbit mereka. Kekurangan kualiti imej boleh menghasilkan imej yang kabur, berpiksel, atau mempunyai resolusi yang rendah. Imej berkualiti rendah ini boleh menyebabkan frustrasi kepada pengguna yang ingin berkongsi imej mereka di platform media sosial atau menggunakannya untuk tujuan peribadi atau profesional. Dalam projek ini, beberapa model rangkaian neural konvolusi mendalam pra-latih iaitu *Enhanced Deep Residual Network (EDSR)*, *Efficient Sub-Pixel Convolutional Neural Network (ESPCN)*, dan *deep Laplacian Pyramid Super-Resolution Network (LapSRN)* digunakan untuk meningkatkan resolusi imej, dan hasil setiap model telah dianalisis dan dibandingkan. *EDSR* dipilih sebagai model terbaik kerana ia telah mencapai nilai purata *PSNR* dan *SSIM* tertinggi dalam pengujian 45 imej iaitu 25.82 dB dan 0.70 masing-masing. Aplikasi pemprosesan imej tradisional sering kali tidak mempunyai keupayaan untuk melaksanakan teknik penambahbaikan imej yang canggih seperti pemindahan gaya neural, yang digunakan untuk mencipta kesan seni pada imej. Fungsi pemindahan gaya neural dicapai dengan mengekstrak gaya imej gaya menggunakan seni bina rangkaian *VGG19* yang pra-latih dan menggunakannya pada imej kandungan untuk mencipta kesan seni pada imej. *VGG19* digunakan kerana ia mencapai *ArtFID* purata tertinggi dalam 30 imej ujian, iaitu 45.97 berbandingkan kepada *MobileNet* dan *ResNet*. Akhirnya, satu aplikasi dibangunkan menggunakan *Flutter* yang menggabungkan semua fungsi di atas.

TABLE OF CONTENTS

PAGE

LIST OF TABLES

PAGE

LIST OF FIGURES

PAGE

LIST OF ABBREVIATIONS

FYP	**-**	**Final Year Project**
CNN	**-**	**Convolution Neural Network**
PSNR	**-**	**Peak Signal to Noise Ratio**
SSIM	**-**	**Structural Similarity Index Measure**
MSE	**-**	**Mean Squared Error**
EDSR	**-**	**Enhanced Deep Residual Networks**
ESPCN	**-**	**Efficient Sub-Pixel Convolutional Neural Network**
LapSRN	**-**	**Deep Laplacian Pyramid Networks**
MCMC	**-**	**Malaysian Communications and Multimedia Commission**
API	**-**	**Application Programming Interface**
VGG19	**-**	**Very Deep Convolutional Networks for Large-Scale Image Recognition**
ResNet	**-**	**Residual Neural Network**
ArtFID	**-**	**Art Fréchet Inception Distance**

CHAPTER 1: INTRODUCTION

1.1 Introduction

In today's digital age, images play a crucial role in communication and self-expression. People take and share hundreds of photos every day on various social media platforms, and the demand for image enhancement tools has never been higher. According to a report by the Malaysian Communications and Multimedia Commission (MCMC) in 2021, approximately 94.8% of Malaysians own a smartphone, indicating a widespread adoption of mobile devices. Among the activities of smartphone users, 74.8% use smartphone to take photos or videos (MCMC Hand Phone Users Survey, 2021). This high penetration rate of smartphones has contributed to a significant increase in the number of photos taken and shared by Malaysians on various social media platforms. Furthermore, a study conducted by Ipsos Malaysia in 2020 revealed that 78% of Malaysians consider the visual quality of images to be essential when sharing them online (Ipsos Malaysia Digital Trends Survey, 2020). This statistic emphasizes the importance of image enhancement tools that can enhance the visual appeal and quality of photos, enabling users to create captivating and engaging content.

In this project, several pretrained deep convolutional neural networks model that are the Enhanced Deep Residual Network (EDSR), Efficient Sub-Pixel Convolutional Neural Network (ESPCN) and deep Laplacian Pyramid Super-Resolution Network (LapSRN) are employed for increasing the resolution of image and the result of each model is analyzed and compared. The performance for each model is measured using metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). PSNR is a widely used metric to evaluate the quality of a reconstructed image. It measures the ratio between the maximum possible power of a signal (usually the original, unaltered image) and the power of the noise or distortion introduced by reconstruction. On the other hand, SSIM is a metric used to assess the similarity between two images. It is designed to capture both structural information and perceived changes in luminance, contrast, and structure (Horé & Ziou, 2013).

Traditional image processing applications often lack the ability to perform advanced image enhancement techniques such as neural style transfer, which can be used to create artistic effects on images. These techniques require a deep understanding of image processing and advanced technical knowledge, which is beyond the scope of most users. The neural style transfer functionality is achieved by extracting the style of the style image using the pretrained image classification network and applying it to the content image to create artistic effects on images. Several pretrained image classification networks such as Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG19), MobileNet and Residual Neural Network (ResNet) network architecture are implemented to compare the quality of the output image between these deep CNN models. The performance for each deep CNN models on neural style transfer technique is evaluated using Art Fréchet Inception Distance (ArtFID) metric. The ArtFID metric is used for assessing the quality of neural style transfer technique and is inspired by the Fréchet Inception Distance (FID) that is used to evaluate the quality of generated images. ArtFID measures the perceptual similarity between the stylized image and a reference image, capturing both the content and style aspects (Wright & Ommer, 2022).

Therefore, there is a need for an application that combines multiple image enhancement tasks in a single, user-friendly platform. This is where ImageGenie comes in. ImageGenie aims to provide users with a unified solution for enhancing image resolution and applying artistic styles to their images.

1.2 Problem Statement

While image processing applications have become more accessible and user-friendly, they still fall short in meeting the needs of users who want to enhance their images without having advanced technical knowledge. Furthermore, low-quality images are a common problem that many people encounter, particularly when taking pictures with their mobile phones. The lack of image quality can result in images that are blurry, pixelated, or have poor resolution. These low-quality images can be frustrating for users who want to share their images on social media platforms or use them for personal or professional purposes. According to a survey conducted by Ipsos in Malaysia (2021), approximately 65% of social media users express dissatisfaction with the quality of images they encounter online. This statistic highlights the need for a solution that addresses image enhancement and empowers users to enhance their low quality images.

With the increasing popularity of social media platforms, people will share images that they think are attractive or interesting to social media. Moreover, many individuals appreciate the beauty of visually captivating artwork, artistic images have the power to evoke emotions, spark creativity, and inspire others. However, for individuals that are lacking the advanced drawing skills, creating such artistic artwork can be quite challenging.

1.3 Objective

This project embarks on the following objectives:

1. To increase the resolution of low quality images using EDSR.

2. To allow users to perform neural style transfer that can compose one super resolved image in the style of another image using VGG19.

1.4 Scope

The scope of the project includes development of a deep learning-based image processing application that can perform a range of image enhancement tasks, such as increasing the resolution of an image and applying neural style transfer.

1.5 Project Significance

By providing a comprehensive solution for image enhancement, ImageGenie will allow people to take their digital images to the next level, making it easier than ever to create and share high-quality, visually appealing content.

1.6 Expected Output

The expected output of the project is a powerful, all-in-one tool for image enhancement that will revolutionize the way people work with their digital images.

1.7 Conclusion

In conclusion, this project will use deep learning models to increase the resolution of low quality images and allow users to perform neural style transfer that can compose one image in the style of another image.

CHAPTER 2: LITERATURE REVIEW AND PROJECT METHODOLOGY

2.1 Introduction

In today's digital age, images have become a vital means of communication and self-expression. People capture and share hundreds of photos daily on various social media platforms, leading to a growing demand for image enhancement tools. Low-quality images are a common problem, particularly when capturing pictures with mobile phones. The lack of image quality can result in blurry, pixelated, or poorly resolved images, causing frustration for users who aim to share their visuals on social media platforms or use them for personal and professional purposes. This project aims to address these challenges by developing an application that employs pretrained deep convolutional neural network models to enhance image resolution and allow users to perform neural style transfer that can compose one image in the style of another image.

This chapter provides a comprehensive review of the literature relevant to this project. The method behind every image enhancement function which is super resolution and neural style transfer is studied. The research paper for every method for the purpose above is read and review. In order to achieve a better result, a thorough review of the references is also conducted.

All of the articles and journals that are related to this project are discussed and the method proposed in these articles is also analyzed in this chapter.

2.2 Facts and Findings

In this section, the detail of this project are concentrated in order to gain a better understanding of the project concept.

2.2.1 Domain

This project focuses on the domain of image enhancement, image processing, and neural networks. It explores various concepts and advancements in these domains to lay the foundation for the project's objectives. Key areas of interest include image resolution enhancement, neural style transfer and deep learning techniques. Understanding these domains is crucial for effectively implementing the project's objectives and achieving high-quality results.

2.2.2 Existing System

Other related projects that are related to this project will also be taken into consideration.

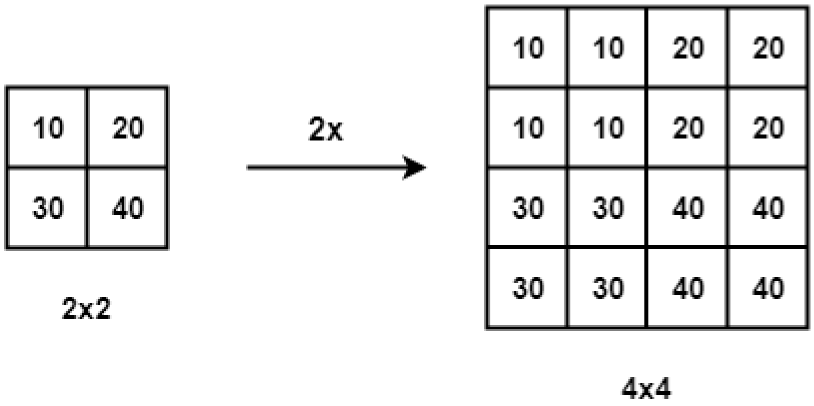
2.2.2.1 Image Super Resolution

Image super resolution not only can be done with deep learning technique, there are also other methods such as interpolation or super resolution forests method. The deep learning method of super resolution is employed in this project because the methods stated above cannot achieve good results of obtaining high quality of super resolved image compare to deep learning method.

(a) Interpolation Upsampling Methods (Ong Si Ci, 2023)

1. Nearest Neighbour Interpolation

This method is straightforward and efficient as it requires minimal calculations. It involves adding pixels based on the intensity values of neighboring pixels. While this approach increases image resolution, it may result in blocky and unnatural-looking images. Figure 2.1 below shows an example of a nearest neighbour interpolation with an upsampling factor of 2 applied to a 2x2 array.



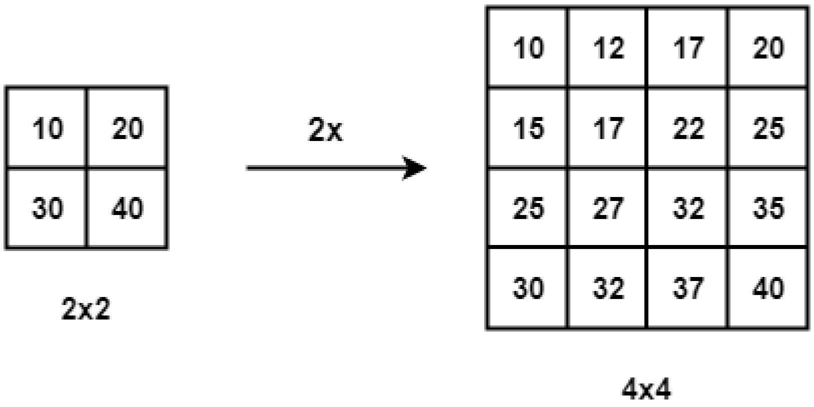
10	20
30	40

10	10	20	20
10	10	20	20
30	30	40	40
30	30	40	40

Figure 2.1: Example of a nearest neighbour interpolation with an upsampling factor of 2

2. Bilinear Interpolation

The bilinear interpolation method employs linear interpolation by calculating the weighted average of the intensity values from the 4 closest neighboring pixels to determine the intensity value of the new pixel. This technique yields smoother results compared to nearest neighbor interpolation, although it may not produce optimal outcomes for edges or sharp transitions. Figure 2.2 below shows the example of a bilinear interpolation with an upsampling factor of 2 applied on a 2x2 array.



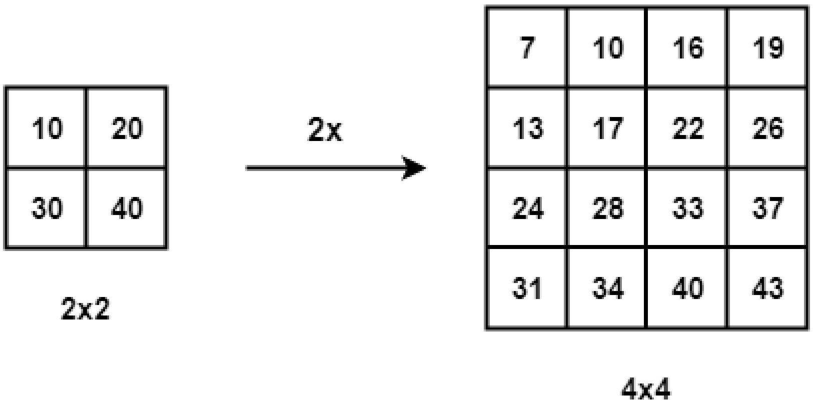
10	20
30	40

10	12	17	20
15	17	22	25
25	27	32	35
30	32	37	40

Figure 2.2: Example of a bilinear interpolation with an upsampling factor of 2

3. Bicubic Interpolation

Bicubic interpolation, similar to the previous methods, utilizes a weighted average of neighboring pixels to generate the output. However, it distinguishes itself by considering a larger neighborhood of 16 pixels (4x4) and employs a different weight distribution calculation. This technique yields sharper images compared to both nearest neighbor and bilinear interpolation and is often used in image editing software. Figure 2.3 below shows the example of a bicubic interpolation with an upsampling factor of 2 applied on a 2x2 array.



10	20
30	40

7	10	16	19
13	17	22	26
24	28	33	37
31	34	40	43

Figure 2.3: Example of a bicubic interpolation with an upsampling factor of 2

(b) Fast and Accurate Image Upscaling with Super-Resolution Forests (Schulter et al., 2015)

The author proposes a method for enhancing the resolution of low-resolution images using a technique called Super-Resolution Forests. The goal is to generate high quality upscaled images that closely resemble the details and sharpness of the original high resolution images.

The method starts by collecting a dataset of low resolution images paired with their corresponding high resolution versions. These pairs are used to train a model called Super-Resolution Forests. The model is designed as a random forest which is a machine learning algorithm that consists of an ensemble of decision trees, where each tree learns to predict the high resolution patch corresponding to a given low resolution patch.

During the training process, the low resolution images are divided into small patches, and the model learns to map these patches to their high resolution counterparts. The Super-Resolution Forests model captures the underlying patterns and relationships between low resolution and high resolution patches, enabling it to make accurate predictions.

Once the model is trained, it can be used to predict the high resolution patches for new low resolution patches. These predicted patches are then combined to reconstruct the upscaled image. To ensure smooth transitions between the patches, a technique called overlap-and-add is employed.

The performance of the Super-Resolution Forests method is evaluated using metrics such as peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM). These metrics assess the accuracy and quality of the upscaled images, comparing them to the ground truth high-resolution images. The effectiveness of the method heavily relies on the quality and diversity of the training dataset. If the dataset does not adequately cover various image characteristics and scenarios, the model's performance may be limited to the specific data it was trained on.

The advantage of Suer-Resolution Forests is that it is fast and efficient. The method utilizes a random forest model, which can efficiently process image patches and predict high resolution patches. This allows for faster super resolution compared to other algorithms. However, the Super-Resolution Forests does not generalize well.

Overall, the paper presents a comprehensive approach to image upscaling using Super-Resolution Forests. By training a random forest model to predict high resolution patches from low resolution patches, the method achieves fast and accurate results.

2.2.2.2 Neural Style Transfer

There are no other technique to do neural style transfer other than deep learning method. Hence, deep learning technique is employed to implement neural style transfer.

2.2.3 Technique

2.2.3.1 Super Resolution

(a) Enhanced Deep Super-Resolution Network (EDSR) (Lim et al., 2017)

The EDSR is a deep learning architecture specifically designed for super-resolving low quality or low resolution images. EDSR aims to enhance the resolution and quality of images by leveraging the power of deep residual networks. EDSR builds upon the concept of residual learning, which is the idea that instead of directly learning the desired mapping from low resolution to high resolution images, it is more effective to learn the residual mapping which is the difference between the low-resolution and high-resolution images. By learning the residual, the network can focus on capturing the high-frequency details and fine-grained features that are necessary for super-resolution.

EDSR utilizes a modified version of the deep residual network (ResNet, SRResNet) architecture. However, to reduce computational complexity, EDSR typically uses a shallower network with fewer layers compared to traditional ResNet models. The batch normalization layers are removed from the original proposed network. This modification allows EDSR to efficiently process and learn the complex mapping from low-resolution to high-resolution. Furthermore, GPU memory usage is also sufficiently reduced since the batch normalization layers consume the same amount of memory as the preceding convolutional layers (Lim et al., 2017). Figure 2.4 below shows the residual blocks of ResNet, SRResNet and EDSR.

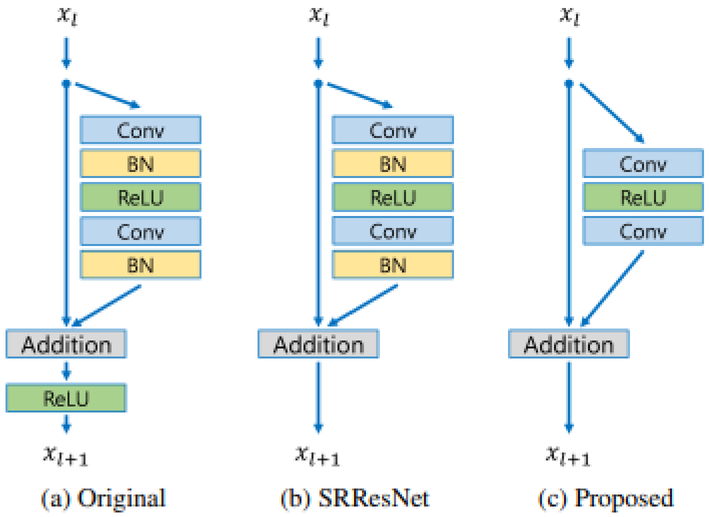


Figure 2.4: Comparison of residual blocks in original ResNet, SRResNet and EDSR (Lim et al., 2017)

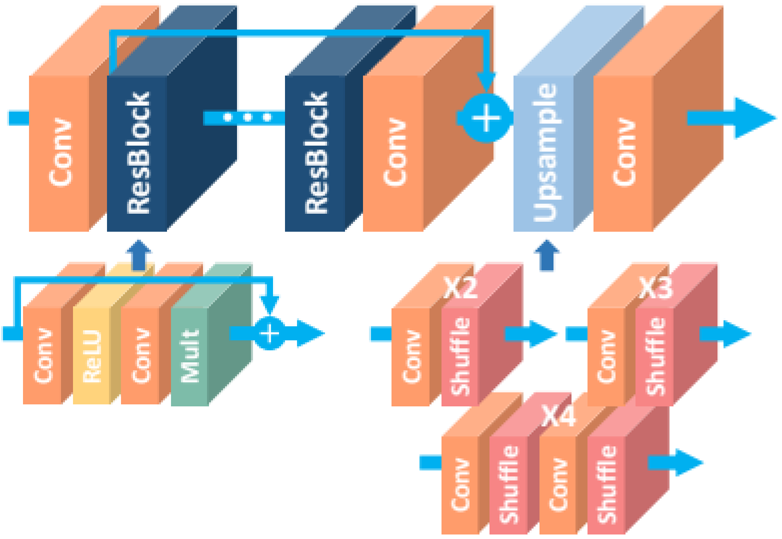


Figure 2.5: The architecture of EDSR (Lim et al., 2017)

Figure 2.5 shows the architecture of the super resolution network EDSR. EDSR incorporates skip connections within its architecture. Skip connections connect the output of one layer to the input of another layer further in the network, allowing the network to directly access and utilize low-level feature information from earlier layers. These skip connections help propagate and preserve information throughout the network, enabling better gradient flow and feature reuse. The network is trained using DIV2K dataset.

During inference, a low-resolution input image is passed through the trained network, which predicts the high-frequency details and generates a super-resolved output image. The network's learned weights and residual mapping capabilities enable it to reconstruct a higher-resolution image with enhanced visual quality and finer details compared to the original low-resolution image.

One of the strengths of EDSR is its deep network architecture. With a large number of convolutional layers, the model can learn complex representations and effectively restore high-frequency details in images. This allows EDSR to achieve high performance in terms of super resolve image quality.

Additionally, EDSR is designed to balance computational efficiency and performance. It can be trained and deployed on modern hardware efficiently, enabling faster training times and real-time or near-real-time inference. This makes EDSR suitable for applications where quick turnaround times are crucial.

Furthermore, EDSR demonstrates flexibility and adaptability. It can handle various super resolution tasks, including increasing the resolution of images with different scales and levels of degradation. The model can be trained on diverse datasets, allowing it to adapt to different image characteristics and capture the specific details of different types of images.

However, EDSR does have some weaknesses that need to be considered. Firstly, the model requires significant computational resources due to its deep architecture. This translates into higher memory and computational requirements during both training and inference, which may limit its usability on resource-constrained devices or in scenarios with limited computational power.

Another weakness of EDSR is that although EDSR is computationally efficient compared to some other super resolution models, achieving real-time performance on resource constrained devices may still be challenging.

(b) Efficient Sub-Pixel Convolutional Neural Network (ESPCN) (Shi et al., 2016)

The ESPCN is a deep learning architecture designed specifically for super-resolution tasks, aimed at enhancing the resolution and quality of low quality or low resolution images. ESPCN leverages sub-pixel convolutional layers to achieve high quality super resolution. The core idea of ESPCN is the use of sub-pixel convolutional layers, also known as pixel shuffle layers or sub-pixel upsampling. These layers aim to upscale the low-resolution input image to a higher resolution by increasing the spatial dimensions while enhancing the details.

Sub-pixel convolution works by transforming low resolution feature maps into high resolution feature maps. It rearranges the feature channels in such a way that each pixel in the low-resolution map contributes to multiple pixels in the high-resolution map. By exploiting this channel-wise rearrangement, the sub-pixel convolutional layer effectively increases the spatial resolution of the feature maps.

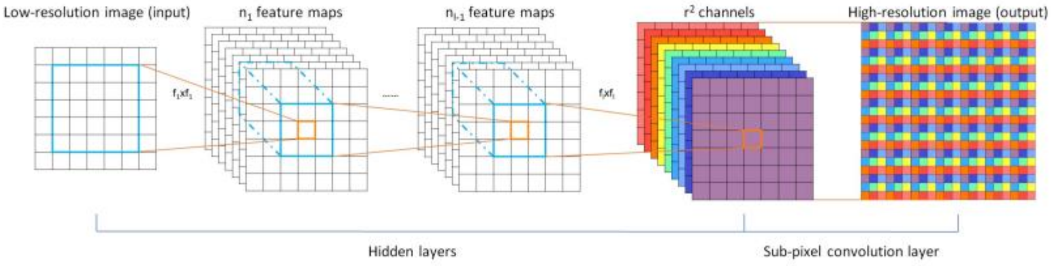


Figure 2.6: ESPCN Network Structure (Shi et al., 2016)

Figure 2.6 shows the network architecture of ESPCN. ESPCN consists of multiple convolutional layers followed by sub-pixel convolutional layers. The initial convolutional layers are responsible for extracting hierarchical features from the low-resolution input image. These features are then passed to the sub-pixel convolutional layers, which perform the upsampling and generate the high-resolution output.

ESPCN aims to provide an efficient solution for super-resolution tasks by using sub-pixel convolutional layers to perform both feature extraction and upsampling in a single network. This design reduces computational complexity compared to traditional super-resolution approaches that involve separate interpolation or upsampling steps. ESPCN has shown promising results in various low-resolution image enhancement tasks and has become a popular choice for real-time applications where efficiency is crucial.

One of the primary strengths of ESPCN is its computational efficiency. The model employs sub-pixel convolutional layers, which allow it to upscale low resolution images to higher resolutions without the need for computationally expensive operations such as interpolation or upsampling. This efficiency makes ESPCN well suited for real time or near real time applications, where quick inference times are crucial.

ESPCN also benefits from its simplicity and lightweight architecture. The model has a relatively small number of parameters compared to other super resolution models, which reduces memory requirements and computational complexity during both training and inference. This simplicity enables easier model deployment and makes ESPCN suitable for deployment on resource constrained devices with limited computational power.

However, ESPCN has certain weaknesses that should be taken into account. One weakness is its limited capability to handle complex image textures and high frequency details. The sub-pixel convolutional layers, while efficient, may struggle to fully restore fine-grained details, resulting in slightly less visually pleasing results compared to more complex models.

(c) Deep Laplacian Pyramid Networks (LapSRN) (Lai et al., 2017)

LapSRN is a deep learning architecture designed for single-image super-resolution, which aims to enhance the resolution and quality of low-quality or low-resolution images. LapSRN leverages the concept of Laplacian pyramid to progressively reconstruct high-resolution images. The Laplacian pyramid is a multi-scale image representation that decomposes an image into different scales or levels, where each level contains the difference between the original image at that scale and a down sampled version of the image. The idea is that the lower levels capture coarse-scale information, while the higher levels capture fine-scale details (Lai et al., 2017). Figure 2.7 below shows the detailed network architecture of LapSRN.

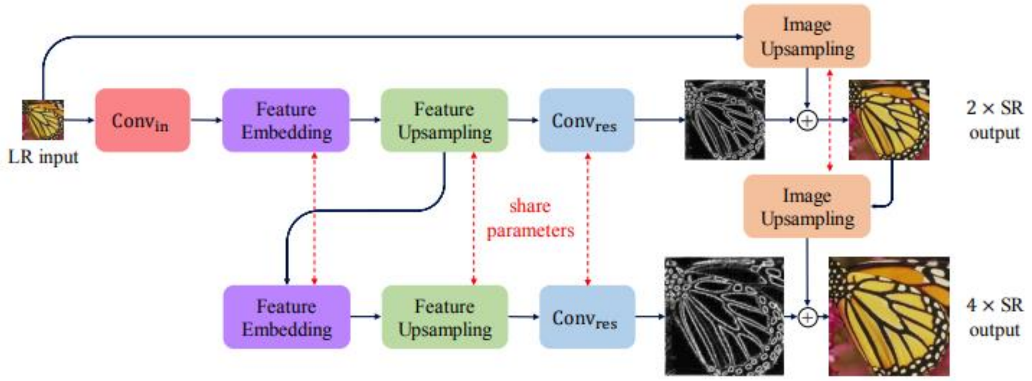


Figure 2.7: Detailed network architecture of LapSRN (Lai et al., 2017)

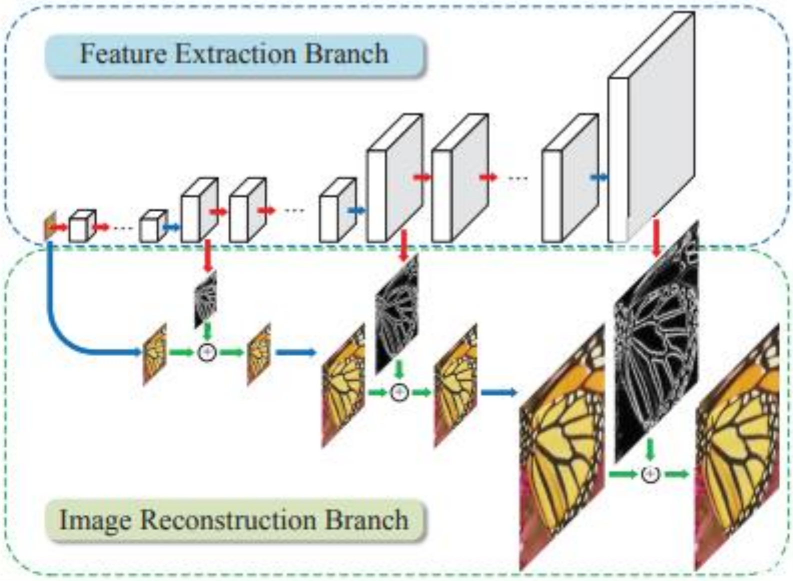


Figure 2.8: Progressive upsampling of LapSRN (Lai et al., 2017)

Figure 2.8 shows the process of upsampling of LapSRN. LapSRN consists of a deep neural network architecture with multiple levels, each corresponding to a different scale of the Laplacian pyramid. The network takes the low-resolution input image as input and progressively upscales it to higher resolutions using the Laplacian pyramid reconstruction approach.

LapSRN utilizes a recursive learning strategy, where the network is trained in a progressive manner from the coarsest scale to the finest scale. At each scale, the network takes the low-resolution input image and generates an enhanced image specific to that scale. The enhanced image is then combined with the upsampled version of the input image from the previous scale to form an input for the next scale. This process continues until the highest resolution is reached.

LapSRN demonstrates excellent scalability in handling varying upscaling factors. The network can generate high resolution images even when faced with significant upscaling ratios. This scalability makes LapSRN suitable for applications that require super resolution of low resolution images with large scale factors, such as enhancing surveillance footage or medical imaging.

However, LapSRN has certain weaknesses that should be considered. One of the weaknesses is its sensitivity to noise and artifacts present in low resolution input images. Since the model relies on the information from the low resolution image to generate the high resolution output, any noise or artifacts present in the input can propagate to the super-resolved image. This sensitivity to noise may limit LapSRN's performance in scenarios where the input images have significant degradation or noise levels.

2.2.3.2 Comparison of Super Resolution Models

PSNR is a widely used metric to evaluate the quality of a reconstructed image. It measures the ratio between the maximum possible power of a signal (usually the original, unaltered image) and the power of the noise or distortion introduced by reconstruction. The PSNR is calculated using the mean squared error (MSE) between the original and reconstructed images and is typically expressed in decibels (dB). A higher PSNR value indicates a higher fidelity and lower distortion in the reconstructed image. The formula to calculate PSNR is shown below (Horé & Ziou, 2013).

$$PSNR = 20log_{10}(\frac{\max_f}{\sqrt{MSE}})$$

Where the MSE (Mean Squared Error) is:

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}|f(i,j) - g(i,j)|^{\,2}$$

SSIM is a metric used to assess the similarity between two images. It is designed to capture both structural information and perceived changes in luminance, contrast, and structure. The final SSIM index value ranges between 0 and 1, with 1 indicating a perfect match and 0 indicating no similarity. Higher SSIM values correspond to images that are visually more similar. The formula to calculate SSIM is shown below (Horé & Ziou, 2013).

$$SSIM(x,y) = \frac{2(\mu_x \mu_y + c1)(2\sigma_{xy} + c2)}{(\mu_x^2 + \mu_y^2 + c1)(\sigma_X^2 + \sigma_y^2 + c2)}$$

2.2.3.3 Neural Style Transfer (Gatys et al., 2015)

Neural style transfer is a technique that combines the content of one image with the style of another image to create a new image that exhibits the content of the first image in the artistic style of the second image. It leverages the power of deep convolutional neural networks (CNNs) to extract and manipulate visual features from images. (Johnson et al., 2016)

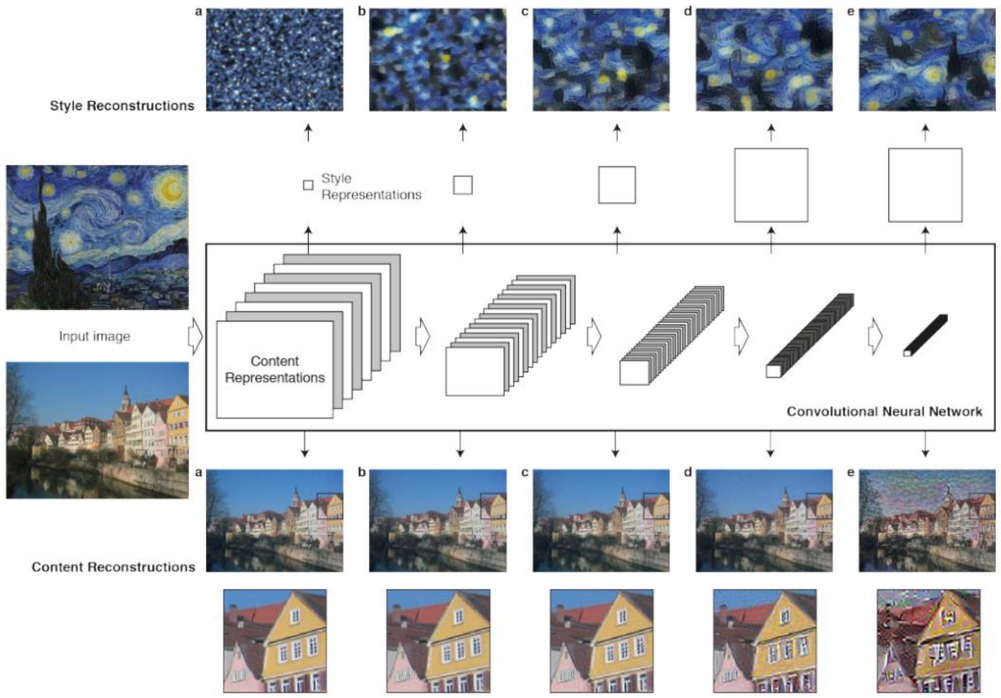


Figure 2.9: Content and Style Representation in CNN (Gatys et al., 2015)

Figure 2.9 shows the content and style image representation in CNN. When CNN are trained on object recognition, they develop a representation of the image that makes object information increasingly explicit along the processing hierarchy. Therefore, along the processing hierarchy of the network, the input image is transformed into representations that increasingly care about the actual content of the image compared to its detailed pixel values. We can directly visualize the information of each layer contains about the input image by reconstructing the image only from the feature maps in that layer. Higher layers in the network capture the high-level content in terms of objects and their arrangement in the input image but do not constrain the exact pixel values of the reconstruction. Reconstructions from the lower layers, on the other hand, simply recreate the precise pixel values of the original image. In higher levels of the network, feature responses are therefore referred to as the content representation.

The author employ a feature space that was initially designed to capture texture information in order to obtain a representation of the style of an input image. The filter responses at each layer of the network serve as the foundation for this feature space. It comprises of the correlations between the different filter responses over the feature map’s whole spatial range. By including the feature correlations of multiple layers, we can captures its texture information and obtain a stable, multi-scale representation of the input image (Gatys et al., 2015).

To extract the content and style information from the input images, pre-trained deep CNN is used. Deep CNN models such as VGG19, MobileNet and ResNet are employed, which have been trained on large-scale image classification tasks (Simonyan & Zisserman, 2014). These networks have learned to capture high-level visual features that are useful for our purposes. The content features are extracted from intermediate layers of the CNN that encode the semantics of the image. These layers capture information about the objects and their spatial arrangement in the content image. For style extraction, CNN is used to compute the Gram matrix for each style image layer.

The Gram matrix represents the correlations between the different features in a given layer. It captures the texture and patterns of the style image by measuring the statistics of the feature maps. Gram Matrix shows the similarity between the filters and is obtained by calculating the dot product between the vectors. Gram matrix is used to measure the style similarity between the style image and the generated output image. By comparing the Gram matrices of different layers between the style image and the output image, the style loss term in the optimization process can be computed. This loss term helps guide the optimization process to generate an output image that matches the style characteristics of the style image.

$$G_j^{\emptyset}(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \emptyset_j(x)_{h,w,c} \emptyset_j(x)_{h,w,c'} .$$

Equation above shows the formula for Gram Matrix(Johnson et al., 2016). The goal of neural style transfer is to find an output image that simultaneously matches the content of the content image and the style of the style image. This is achieved by defining a loss function that measures the difference between the content and style representations of the output image and the target images. The loss function typically consists of two terms: the content loss and the style loss. The content loss measures the difference in feature responses between the content image and the output image. The style loss quantifies the difference in Gram matrices between the style image and the output image at multiple layers.

The optimization process aims to minimize the defined loss function with respect to the pixel values of the output image. Starting with a random noise image or a copy of the content image, an iterative optimization algorithm such as gradient descent is employed to update the pixel values. The optimization process adjusts the image to minimize the content and style differences simultaneously, gradually transforming it to achieve the desired style transfer effect. By iteratively optimizing the pixel values, the output image gradually converges towards a visually pleasing image that combines the content of the content image with the style of the style image.

The ArtFID metric is used for assessing the quality of neural style transfer technique and is inspired by the Fréchet Inception Distance (FID) that is used to evaluate the quality of generated images. ArtFID measures the perceptual similarity between the stylized image and a reference image, capturing both the content and style aspects. The authors conduct extensive experiments using various neural style transfer algorithms and datasets that are labeled by artists to demonstrate the effectiveness of the ArtFID metric (Wright & Ommer, 2022). They compare ArtFID against other metrics, showcasing its ability to align better with human perception of style transfer quality. Equation below depicts the formula to calculate ArtFID (Wright & Ommer, 2022).

$$ArtFID\left(X_g, X_c, X_s\right) = (1 + \frac{1}{N}\sum_{i=1}^{N} d\left(X_c^{(i)}, X_g^{(i)}\right)) \cdot (1 + FID\left(X_s, X_g\right))$$

2.3 Project Methodology

This project adopts the Agile methodology. Agile is an iterative development that emphasizes continuous incremental delivery of the application. At the same time, this allows action to incorporate any feedback to improve the application throughout the development process and adapts to any change in requirement. Figure 2.10 shows the Agile Methodology.

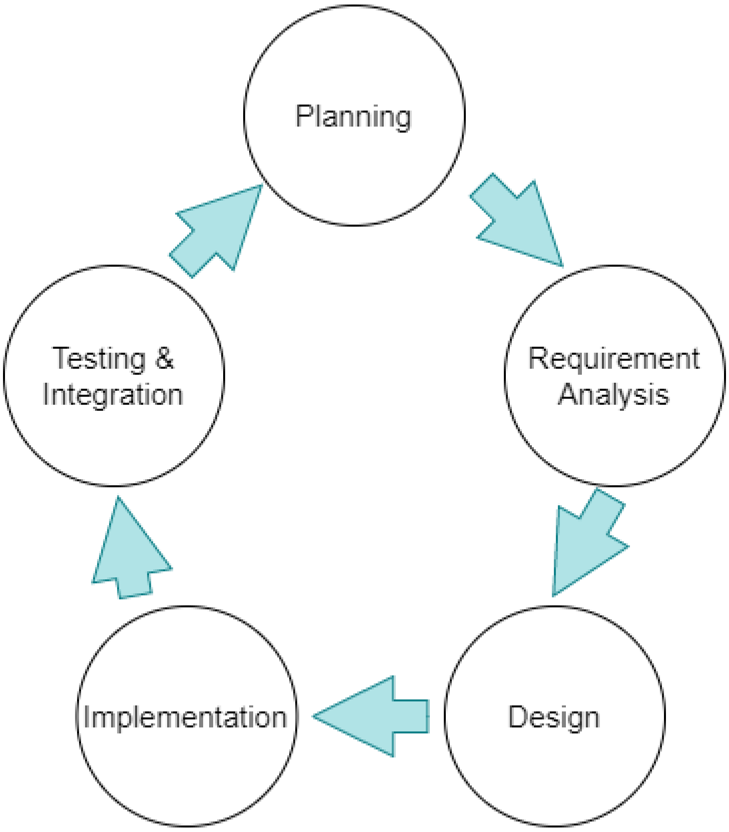


Figure 2.10: Agile Methodology

2.3.1 Planning Phase

The project planning phase includes choosing the project title, identify project objectives, scopes, milestones, and developing the Gantt Chart. The Gantt Chart serves as a visual representation of the project timeline, allowing for effective scheduling and task allocation. The functionality of the system is determined and the targeted operating system that the system will be running is also decided. The overall design architecture and the technology that are going to be used in the system is also identified.

2.3.2 Requirement Analysis

For requirement analysis, thorough research is done to analyze similar current applications and their advantages and limitations. Additionally, various methods applicable to the project's objectives are studied and compared to identify the most suitable approach. The respective research papers for all the deep learning models is studied and investigated in order to fully understanding the architecture and theory behind every deep learning models that enable it to perform respective image enhancement operations.

2.3.3 Design

This phase is the design of the project. For the backend part, Django is chosen as the framework that creates the Tensorflow deep learning api for processing the image. Django was chosen because of its modularity that makes it easy to maintenance when the code base is getting large. The api will be consumed by frontend or the mobile application using Flutter. Flutter was chosen because it is cross-platform, thus enables the code to be shared across multiple platforms (Android, IOS and window). Flutter is also easy to learn, and it provides material design applications which are attractive to the users. Moreover, Flutter can build applications that are as high-performance as native applications (Tashildar et al., 2020). Figure 2.11 shows the architecture of the ImageGenie application.

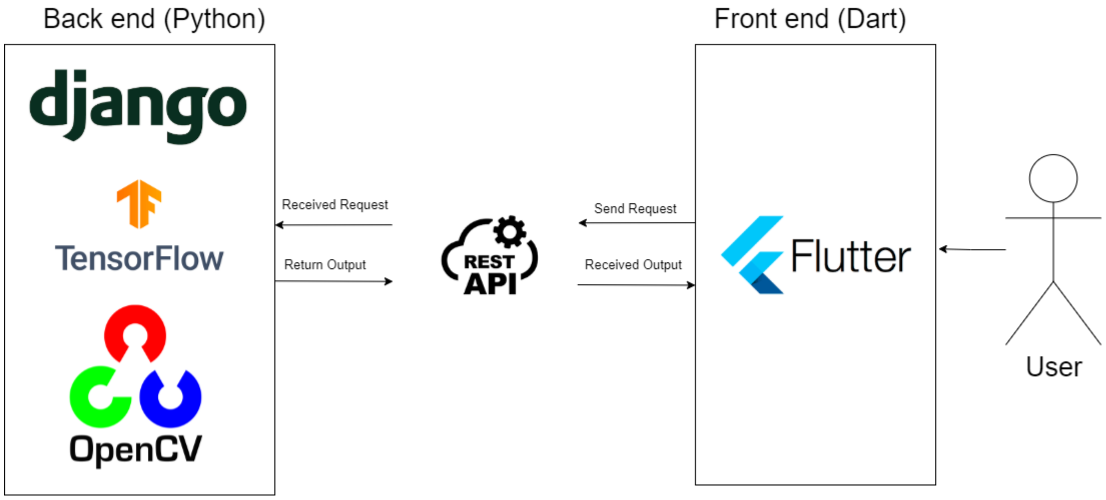


Figure 2.11: The design of the ImageGenie application

2.3.4 Implementation

For implementation, the mobile application as well as the Django backend is developed. The user can upload their images through the mobile application to the backend to carry out super resolution for the image and compose the image in the style of another image. The image received by the Django will pass to the Tensorflow and OpenCV deep learning api to carry out the task.

2.3.5 Testing and Integration Phase

In this stage, it is crucial to ensure that the system runs smoothly and meets the expected requirements. The integration of the backend with the frontend is tested to prevent request time out problem. The result of each functionality provided by the api will be tested to ensure it is working and the processed image will be sent back to the mobile application successfully. Any issues or bugs identified during this phase are addressed, with the aim of ensuring a smooth and reliable user experience. In cases where the system encounters difficulties, the implementation phase is revisited to identify and rectify potential problems.

2.4 Project Requirements

This section will include a list of the software and hardware that were utilized to complete this project.

2.4.1 Software Requirement

The list of software needed to complete this project:

- Microsoft Office 2019 – Software for writing the report
- Visual Studio Code – Software used to develop the application and execute analysis process to the deep learning models.
- Draw.io – Software used to produce flowcharts and diagrams.

2.4.2 Hardware Requirement

The hardware used to complete this project:

- Laptop

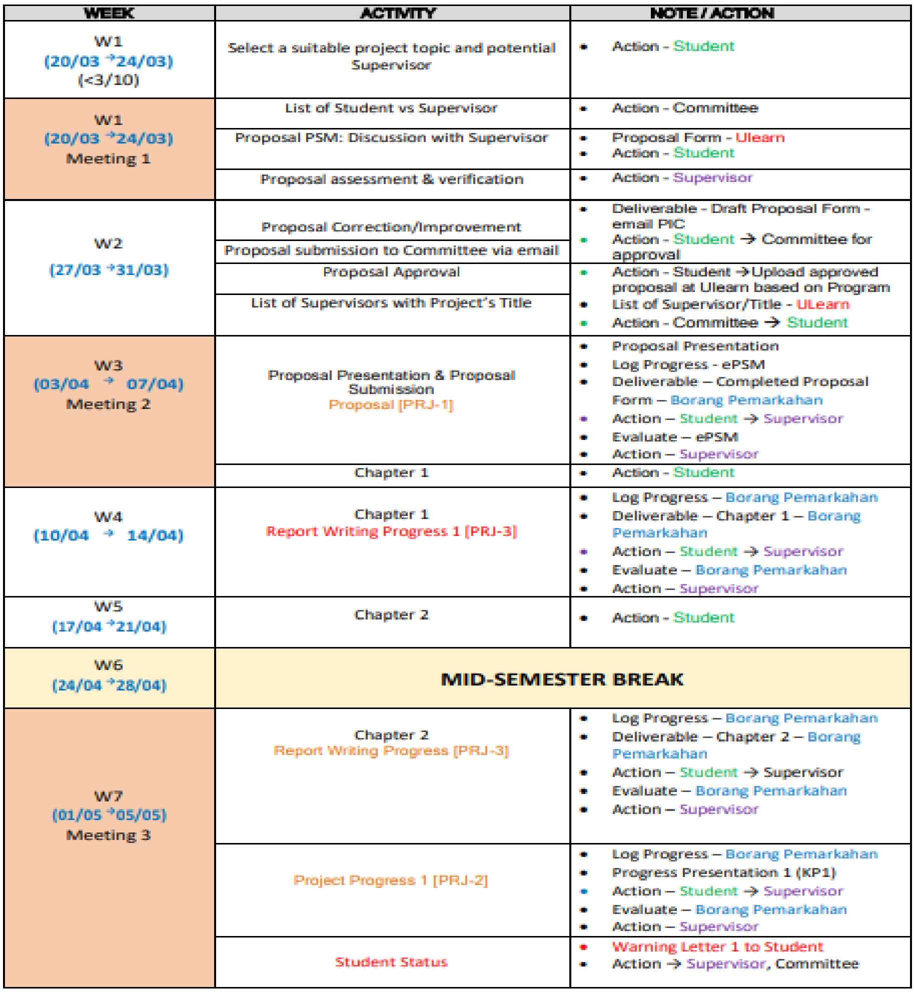
   i. Intel Core i5-1035G1 with NVIDIA GeForce MX350

   ii. Windows 10 64-bit operating system

2.5 Project Schedule and Milestones

This section shows the project's timeline and milestones from the proposal phase until the final presentation. Table 2.1 shows the project milestones while Table 2.2 shows the Gantt chart of this project.

Table 2.1: Project Milestones



WEEK	ACTIVITY	NOTE / ACTION
W1 (20/03 →24/03) (<3/10)	Select a suitable project topic and potential Supervisor	• Action - Student
W1 (20/03 →24/03) Meeting 1	List of Student vs Supervisor	• Action - Committee
	Proposal PSM: Discussion with Supervisor	• Proposal Form - Ulearn • Action - Student
	Proposal assessment & verification	• Action - Supervisor
W2 (27/03 →31/03)	Proposal Correction/Improvement	• Deliverable - Draft Proposal Form - email PIC • Action - Student → Committee for approval
	Proposal submission to Committee via email	
	Proposal Approval	• Action - Student →Upload approved proposal at Ulearn based on Program • List of Supervisor/Title - ULearn • Action - Committee → Student
	List of Supervisors with Project's Title	
W3 (03/04 → 07/04) Meeting 2	Proposal Presentation & Proposal Submission Proposal [PRJ-1]	• Proposal Presentation • Log Progress - ePSM • Deliverable – Completed Proposal Form – Borang Pemarkahan • Action – Student → Supervisor • Evaluate – ePSM • Action – Supervisor
	Chapter 1	• Action - Student
W4 (10/04 → 14/04)	Chapter 1 Report Writing Progress 1 [PRJ-3]	• Log Progress – Borang Pemarkahan • Deliverable – Chapter 1 – Borang Pemarkahan • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
W5 (17/04 →21/04)	Chapter 2	• Action - Student
W6 (24/04 →28/04)	MID-SEMESTER BREAK	
W7 (01/05 →05/05) Meeting 3	Chapter 2 Report Writing Progress [PRJ-3]	• Log Progress – Borang Pemarkahan • Deliverable – Chapter 2 – Borang Pemarkahan • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
	Project Progress 1 [PRJ-2]	• Log Progress – Borang Pemarkahan • Progress Presentation 1 (KP1) • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
	Student Status	• Warning Letter 1 to Student • Action → Supervisor, Committee

W8 (08/05 → 12/05)	Chapter 3	• Action - Student
W9 (15/05 → 19/05)	Chapter 3 Report Writing Progress 1 [PRJ-3]	• Log Progress – Borang Pemarkahan • Deliverable – Chapter 3 – Borang Pemarkahan • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
W10 (22/05 → 26/05) Meeting 4	Chapter 4	• Action - Student
	Project Progress 2 [PRJ-4]	• Log Progress – Borang Pemarkahan • Progress Presentation 2 (KP2) • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
	Student Status	• Warning Letter 2 to Student • Action – Supervisor, Committee
W11 (29/05 → 02/06)	Chapter 4 Report Writing Progress 2 [PRJ-5]	• Log Progress – Borang Pemarkahan • Deliverable – Chapter 4 – Borang Pemarkahan • Action – Student → Supervisor • Evaluate – Borang Pemarkahan • Action – Supervisor
	PSM1 Draft Report preparation	• Action – Student
	Determination of Student Status (Continue/Withdraw)	• Submit Student status to PSM/PD Committee • Action – Supervisor → Committee
W12 & W13 (05/06 → 16/06) Meeting 5	PSM1 Draft Report preparation	• Action - Student → Supervisor
W14 (19/06 → 23/06)	PSM1 Draft Report submission to SV & Evaluator Report Evaluation [PRJ6] [PRJ-10]	• Log Progress – Borang Pemarkahan • Deliverable – Complete PSM1 Draft Report – Borang Pemarkahan • Action – Student → Supervisor, Evaluator • Evaluate – Borang Pemarkahan • Action – Supervisor
	Schedule the presentation	• Presentation Schedule - ULearn • Action - Committee
W15 (26/06 → 30/06) FINAL PRESENTATION	Demonstration Supervisor [PRJ-6] Evaluator [PRJ-7]	• Log Record – Borang Pemarkahan • Action – Student • Evaluate – Borang Pemarkahan • Action – Supervisor, Evaluator
	Presentation Skill [PRJ-8]	• Log Record – Borang Pemarkahan • Action – Student • Evaluate – Borang Pemarkahan • Action – Evaluator
W16 (03/07 → 07/07) REVISION WEEK	**REVISION WEEK** Correction on the draft report based on the Supervisor and Evaluator's comments during the final presentation session. Do an EoS Survey online form.	• Deliverable – PSM1 Report – ULearn • Action – Student→ Committee • Deliverable – EoS Survey – Online Form • Action – Student
	Complete overall marks to Committee	• Deliverable: Overall Evaluation PSM 1 – Borang Pemarkahan • Action – Supervisor, Evaluator → Committee
W17 & W18 (10/07 → 21/07)	**FINAL EXAMINATION WEEKS**	

Table 2.2: Gantt Chart

Activity/Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Proposal	🟧	🟧	🟧												
Project Progress 1				🟥	🟥	🟥	🟥	🟥	🟥						
Report Writing Progress 1			🟨	🟨	🟨	🟨	🟨	🟨	🟨						
Project Progress 2										🟩	🟩	🟩	🟩	🟩	
Report Writing Progress 2											🟦	🟦	🟦	🟦	
Demonstration															🟪
Report Evaluation															🟪
Presentation															🟪

2.6 Conclusion

In conclusion, this chapter is important because we need to conduct research and acquire as much information as possible in order to better understand the project. Other than that, in this chapter, the literature review and project methodology have been discussed, providing an overview of the existing system, techniques, project requirements, and the selected approach. The chapter sets the foundation for the subsequent chapters, providing a comprehensive understanding of the project's background, research context, and implementation approach.

CHAPTER 3: ANALYSIS

3.1 Introduction

The analysis phase of this project plays a crucial role in understanding and determining the requirements to complete this project from the analysis that conducted. By conducting a thorough analysis, we can reduce the overall problems and difficulties that we will face. The analysis is broken down to two sections which are problem analysis and requirement analysis. This section provides an in-depth overview of how the analysis phase will be developed to address the objectives.

There are many deep learning models implemented in this project. EDSR, ESPCN and LapSRN can be used to increase the resolution of low quality images. The models used to super resolve the low quality image is from the deep Convolution Neural Network (CNN) family. They each have their own unique architecture that produces different results. The Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) metrics are calculated to evaluate the performance of the deep learning models to increase the quality of poor image. Based on the metrics, the models are compared, and the best model is chosen to be implemented in the system. For neural style transfer, deep CNN models which are VGG19, MobileNet and ResNet are employed to capture the high level content and style features that are useful for our purpose. The ArtFID metric is used for assessing the quality of neural style transfer technique. The best deep CNN model was chosen to implement in system later.

3.2 Problem Analysis

To gain insights into the current scenario of ImageGenie, an extensive investigation was conducted into existing methods and tools. The primary focus was to understand how these methods operate and the limitations they face. The problem for this project is how to successfully increase the resolution of low quality images. Image super resolution has gained increasing attention for decades. Super resolution aims to recreate a high resolution image from a low resolution image. Recently, deep neural networks provide significantly improved performance in terms of PSNR in the super resolution problem (Lim et al., 2017). However, such network exhibit limitations in terms of architecture optimality. The reconstruction performance of the neural network models is sensitive to minor architectural changes. EDSR, ESPCN and LapSRN can be used to increase the resolution of low quality images. The models used to super resolve the low quality image is from the deep Convolution Neural Network (CNN) family. They each have their own unique architecture that produces different results. The Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) metrics are calculated to evaluate the performance of the deep learning models to increase the quality of poor image. Based on the metrics, the models are compared, and the best model is chosen to be implemented in the system. Figure 3.1 illustrate the flowchart for the comparison of the performance of super resolution models EDSR, ESPCN and LapSRN.

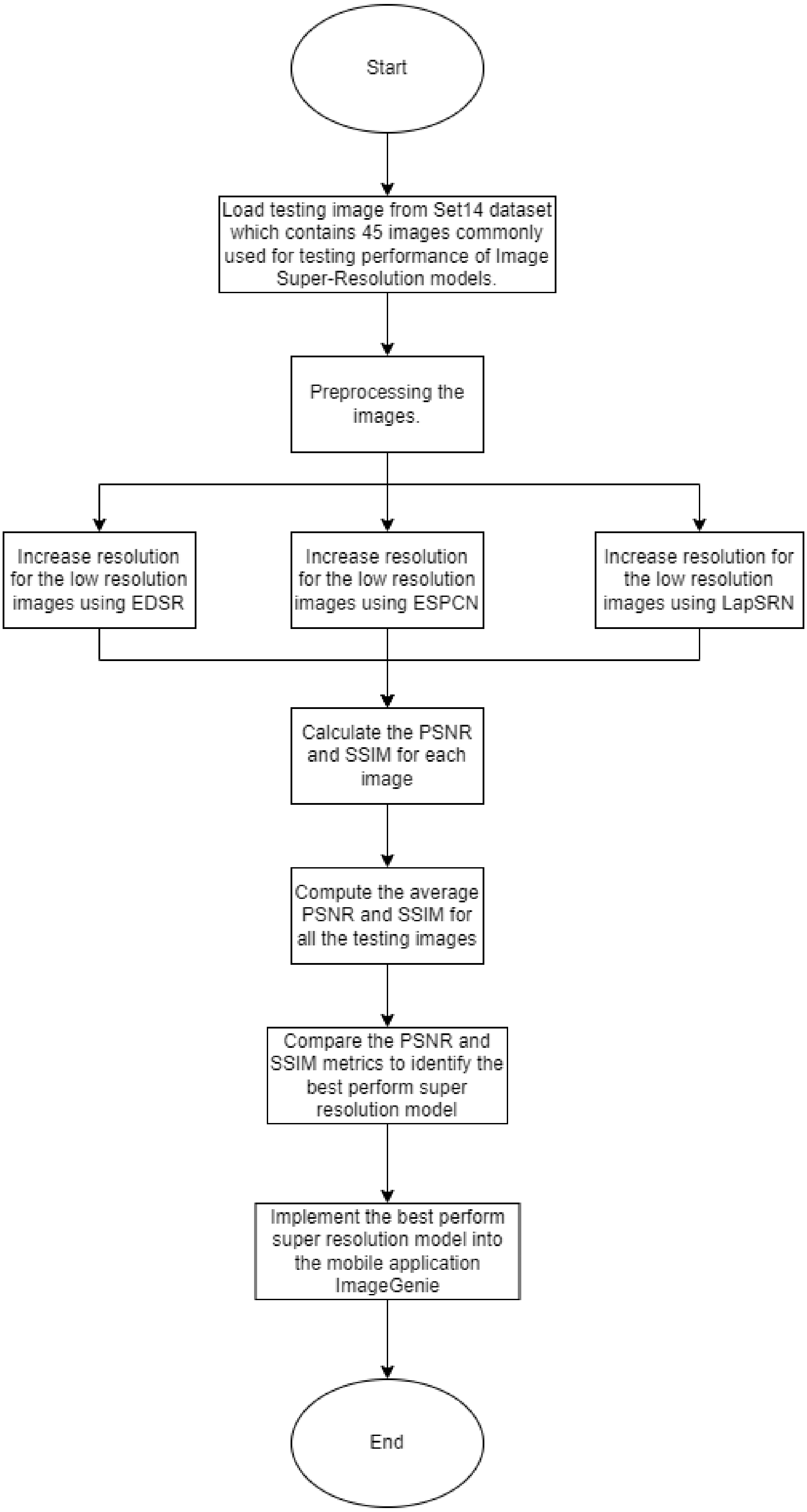

Start
Load testing image from Set14 dataset which contains 45 images commonly used for testing performance of Image Super-Resolution models.
Preprocessing the images.
Increase resolution for the low resolution images using EDSR
Increase resolution for the low resolution images using ESPCN
Increase resolution for the low resolution images using LapSRN
Calculate the PSNR and SSIM for each image
Compute the average PSNR and SSIM for all the testing images
Compare the PSNR and SSIM metrics to identify the best perform super resolution model
Implement the best perform super resolution model into the mobile application ImageGenie
End

Figure 3.1: Flowchart for the Super Resolution Models Performance Comparison

For the implementation of neural style transfer on image, the neural representation of the content and style image need to be extracted clearly so that the reconstructed image can have the style of style image but do not lose their actual content. This can be accomplished by utilizing deep convolutional neural networks (CNN) that are trained on object recognition. Deep CNN models which are VGG19, MobileNet and ResNet are employed to capture the high level content and style features that are useful for our purpose. The ArtFID metric is used for assessing the quality of neural style transfer technique. The best deep CNN model was chosen to be implemented in the system. Figure 3.2 shows the flowchart for the comparison of the performance of deep CNN models in neural style transfer.

Furthermore, statistical and mathematical formulations were employed to validate the effectiveness of the deep convolutional neural networks models used in this project. Computational analysis was performed to evaluate the performance and accuracy of these models.

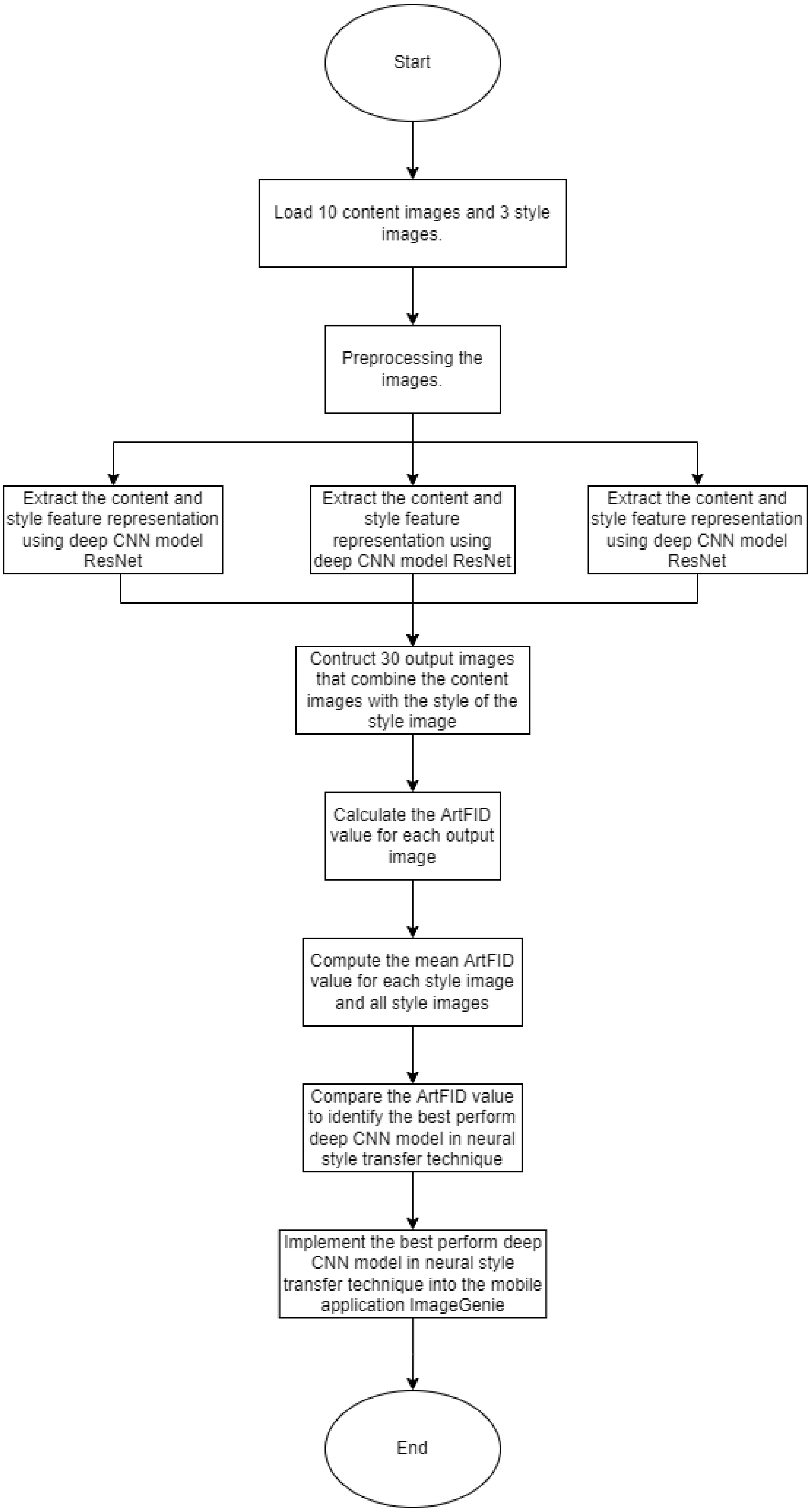

Start
Load 10 content images and 3 style images.
Preprocessing the images.
Extract the content and style feature representation using deep CNN model ResNet
Extract the content and style feature representation using deep CNN model ResNet
Extract the content and style feature representation using deep CNN model ResNet
Contruct 30 output images that combine the content images with the style of the style image
Calculate the ArtFID value for each output image
Compute the mean ArtFID value for each style image and all style images
Compare the ArtFID value to identify the best perform deep CNN model in neural style transfer technique
Implement the best perform deep CNN model in neural style transfer technique into the mobile application ImageGenie
End

Figure 3.2: Flowchart for the deep CNN Models in Neural Style Transfer Performance Comparison

3.3 Requirement Analysis

This section will discuss the requirements that must be met in order to complete this project. Data Requirement, Functional Requirement, Non-Functional Requirement, and Others Requirement will be discussed.

3.3.1 Data Requirement

In this section, we identify the data requirements for this project. The system should have the capability to accept input images from users. These images can be uploaded through the application's user interface, providing a seamless experience for users to select images for enhancement. Additionally, the system needs to store various types of data in the server temporarily. This includes original images, output images and style images used for artistic effects.

The data used for testing the super resolution models are taken from publicly available benchmark dataset. The Set14 dataset is a dataset that consisting of 45 images commonly used for testing performance of image super resolution models (Zeyde et al., 2012). It consists of low resolution image as well as the respective high resolution image version that are useful to evaluate the quality of the reconstructed image from image super resolution process.

Figure 3.3: Example 1 for Low Resolution Image Dataset

Figure 3.4: Example 2 for Low Resolution Image Dataset

On the other hand, the testing on the performance of neural style transfer CNN models ResNet, MobileNet and VGG19 use 10 custom content images with 3 style images. Each deep CNN models powered neural style transfer technique will experiment in processing 10 content images with 3 style images which will result in 30 output transfered image which combine the content images with the style of style images.

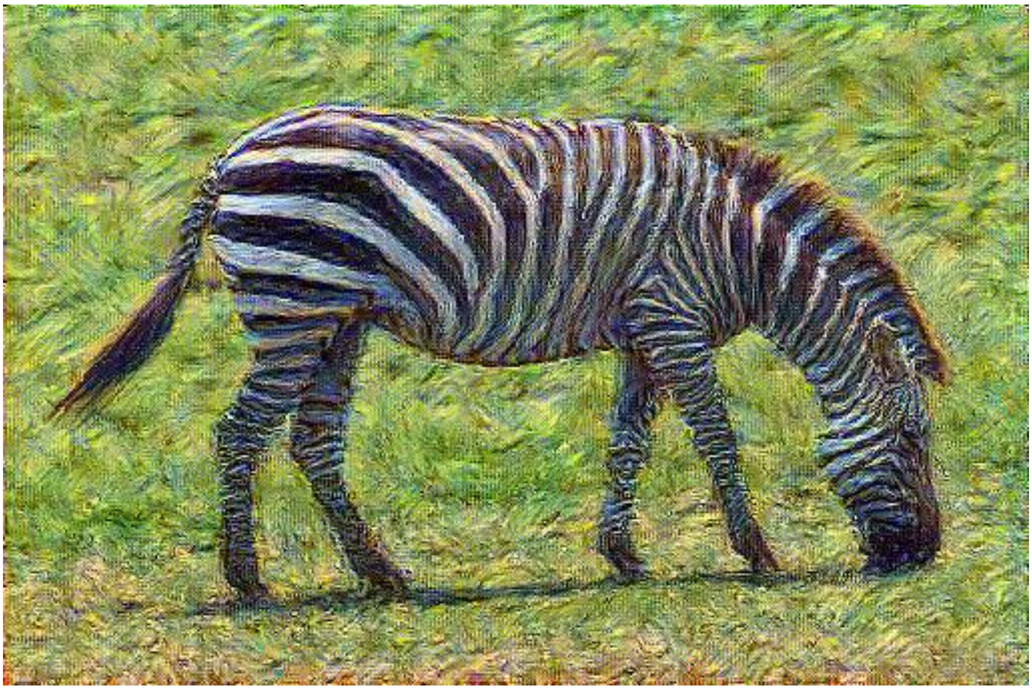


Figure 3.5: Output Image Example Using MobileNet

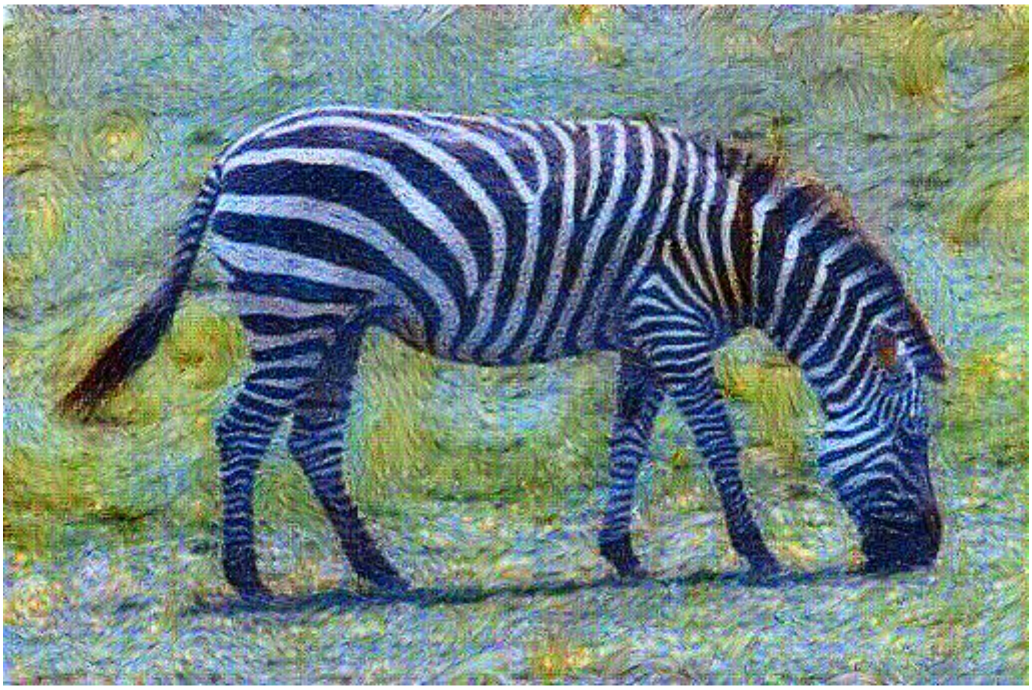


Figure 3.6: Output Image Example Using ResNet

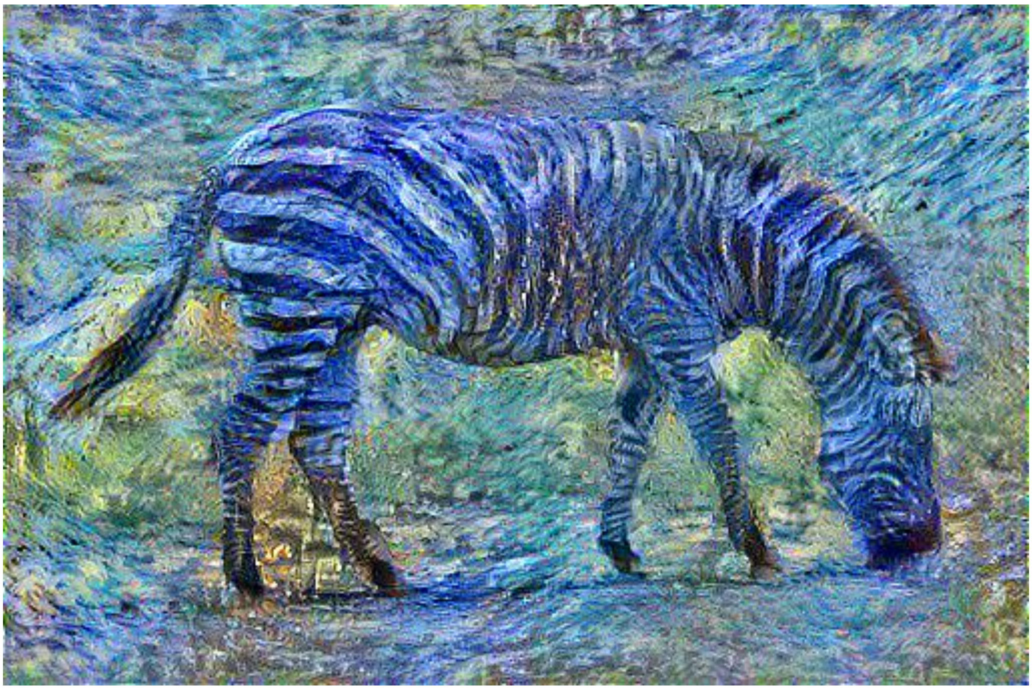


Figure 3.7: Output Image Example Using VGG19

3.3.2 Functional Requirement

The functional requirements define the core functionalities of the image enhancement techniques. The system should incorporate several pretrained deep convolutional neural networks models, namely the Enhanced Deep Residual Network (EDSR), Efficient Sub-Pixel Convolutional Neural Network (ESPCN), and deep Laplacian Pyramid Super-Resolution Network (LapSRN). These models will be utilized to enhance the resolution of images.

Additionally, the system should incorporate the functionality of neural style transfer, allowing users to apply artistic effects to their images. This is achieved by extracting the style of a reference image using the VGG19, MobileNet and ResNet network architecture, which is a pretrained image classification network. The extracted style is then applied to the content image, resulting in an image with the desired artistic effect.

3.3.3 Non-functional Requirement

In addition to the functional requirements, non-functional requirements specify how well the system should perform its intended functions. PSNR and SSIM metrics are used to evaluate the quality of a super resolved image to ensure that the enhanced images are visually appealing and maintain a high level of image quality. The inference time is also recorded to observe the performance of the models. The ArtFID metric is used for assessing the quality of neural style transfer technique and is inspired by the Fréchet Inception Distance (FID) that is used to evaluate the quality of generated images. ArtFID measures the perceptual similarity between the stylized image and a reference image, capturing both the content and style aspects.

3.3.4 Others Requirement

This section will list and discuss the detailed analysis of software and hardware requirements.

3.3.4.1 Software Requirements

Table 3.1: Software Requirements

Software	Usage
Microsoft Word 2019	Writing proposal, report and other documentations for this project.

Microsoft PowerPoint	Preparation for the presentation slide.
Microsoft Visual Studio Code	Develop the application, test and evaluate the models
Draw io	Draw diagram, flowchart and charts
Python	The deep learning technique is implemented using Tensorflow and a server is built using Django to integrate with mobile application
Flutter	Used to create mobile application

3.3.4.2 Hardware Requirements

Table 3.2: Hardware Requirements

Hardware	Usage
Processor	Intel Core i5-1035G1 with NVIDIA GeForce MX350
System Type	64 bit Windows 10 operating system
Installed RAM	12GB
Memory Type	DDR4

3.4 Conclusion

In conclusion, Chapter 3 of the report provided a comprehensive analysis of the image enhancement techniques. The requirement analysis outlined the data, functional, non-functional, and other requirements necessary for the development of the system. By thoroughly analyzing and defining these requirements, we can proceed with the design and implementation phases effectively.

CHAPTER 4: DESIGN

4.1 Introduction

The design of the system will be discussed in this chapter. This chapter defines the results of the analysis of the preliminary design and the result of the detailed design. The chapter aims to define the results of our design process and lay the foundation for the implementation phase, ensuring that our system is well-structured and efficient.

4.2 High Level Design

The goal of this project is to develop a mobile application that combines several image enhancement techniques. These techniques include super resolving a low quality image and perform neural style transfer that can compose one image in the style of another image. The image enhancement method will be done on the Django server which used the python Tensorflow library to load the related models and perform respective functions based on the action from the mobile application.

4.2.1 System Architecture

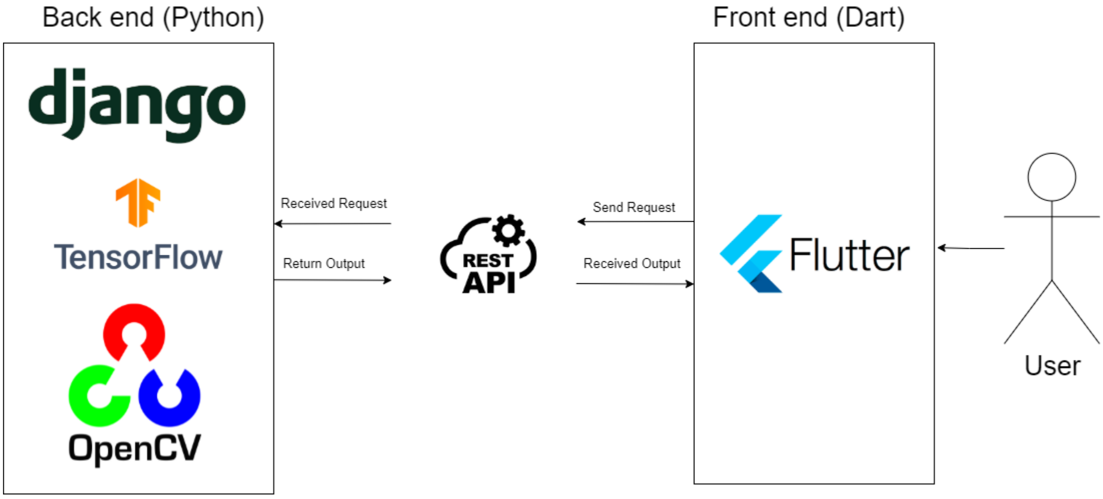


Figure 4.1: System architecture of ImageGenie

The system architecture is shown in Figure 4.1 where the frontend part is developed using Flutter which is a framework to create cross platform applications. The user will upload the image through the mobile application and the request will be sent to the backend using Rest API. The image will then load using OpenCV and processed using Tensorflow in the backend based on the choice of user in which image enhancement they want to do.

4.2.2 User Interface Design

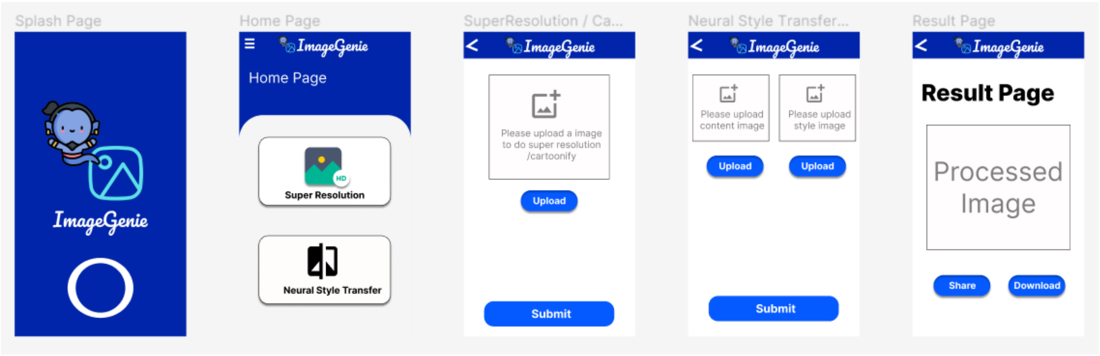


Figure 4.2: User interface design for ImageGenie using Figma

The design of the user interface of this project is depicted in Figure 4.2. The project's interface will be made using the Flutter framework. The application will have splash page, home page, super resolution page, neural style transfer page and result page. Each page is built using the Flutter Scaffold widget which provide the basic starting point that provide material design user interface out of the box. Then, all the component inside the application such as loading circle, button, text, appbar is built using widget from Flutter. The loading widget is built in widget which provide loading effect and is implemented in splash page and result page. All the button is implemented using custom widget in Flutter which we built the button from scratch using combination of many widgets such as container widget, text widget, material widget and so on. The name of the application at the appbar use GoogleFont “Pacifico”. The transition between pages is built using Hero widget which bring the smooth transition from one page to another page. The state management library used in this application is the Provider state management library which can manage the state of the application and can use it at any page of the application to provide functionality like accessing the input image and processed image in more than one pages.

4.2.2.1 Navigation Design

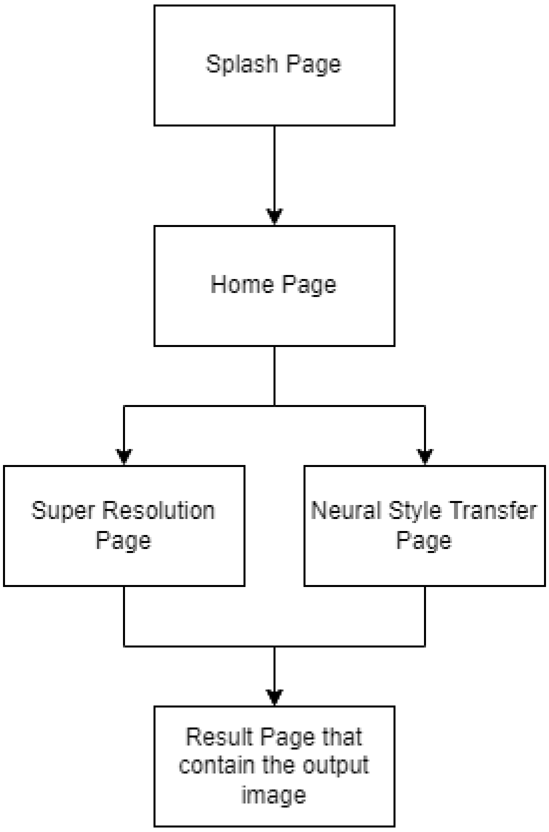

Splash Page
Home Page
Super Resolution Page
Neural Style Transfer Page
Result Page that contain the output image

Figure 4.3: Navigation design of ImageGenie

Figure 4.3 above shows the navigation flow of the application of this project, ImageGenie. Firstly, when the user enters the application, a splash page will show to welcome the user. Then after 2 seconds, the splash page will pop out of the navigation stack and the home page is pushed into the navigation stack to let user enter the home page. In the home page, there are two buttons that allow the users to choose which page they want to go to perform respective image enhancement technique. Finally, the output image will be shown on the result page.

4.2.2.2 Input Design

The type of inputs that will be accepted by the super resolution page and neural style transfer page are image. Image picker will be implemented to let users choose an image to upload from their gallery.

4.2.2.3 Technical Design

The technical design phase involves refining the selected artificial intelligence techniques and algorithms that we have chosen for our project. We will provide a detailed description of each technique and algorithm, highlighting their functionalities and benefits.

For image resolution enhancement, after doing the comparison on the performance of the pretrained deep convolutional neural network models using PSNR and SSIM metrics, we will utilize EDSR models since it provides the best result for image super resolution.

Additionally, we will integrate the VGG19, MobileNet and ResNet network architecture, a pretrained image classification network, for neural style transfer. This technique involves extracting the style of a reference image and applying it to the content image, resulting in artistic effects and stylized images.

Different pretrained image classification network, for neural style transfer which are ResNet, MobileNet and VGG19 were employed and the performance or quality was measured using evaluation metric ArtFID. The ArtFID metric is used for assessing the quality of neural style transfer technique.

Each CNN models powered neural style transfer technique will experiment in processing 10 content images with 3 style images which will result in 30 output image which combine the content images with the style of style images. The 30 images will then be evaluated using ArtFID metric. ArtFID metric with higher values indicating better performance.

In the experiment, VGG19 appears to be the most consistent and effective model among the three models, as it has the highest mean ArtFID value in the neural style transfer technique done toward 10 content images and three style images.

4.2.2.4 Output Design

The output design focuses on defining the types of output generated by our system. We will provide detailed reports, including image enhancement techniques, comparisons of different models, and analysis of the results.

4.3 Detailed Design

In this section, we will describe about the software design.

4.3.1 Software Design

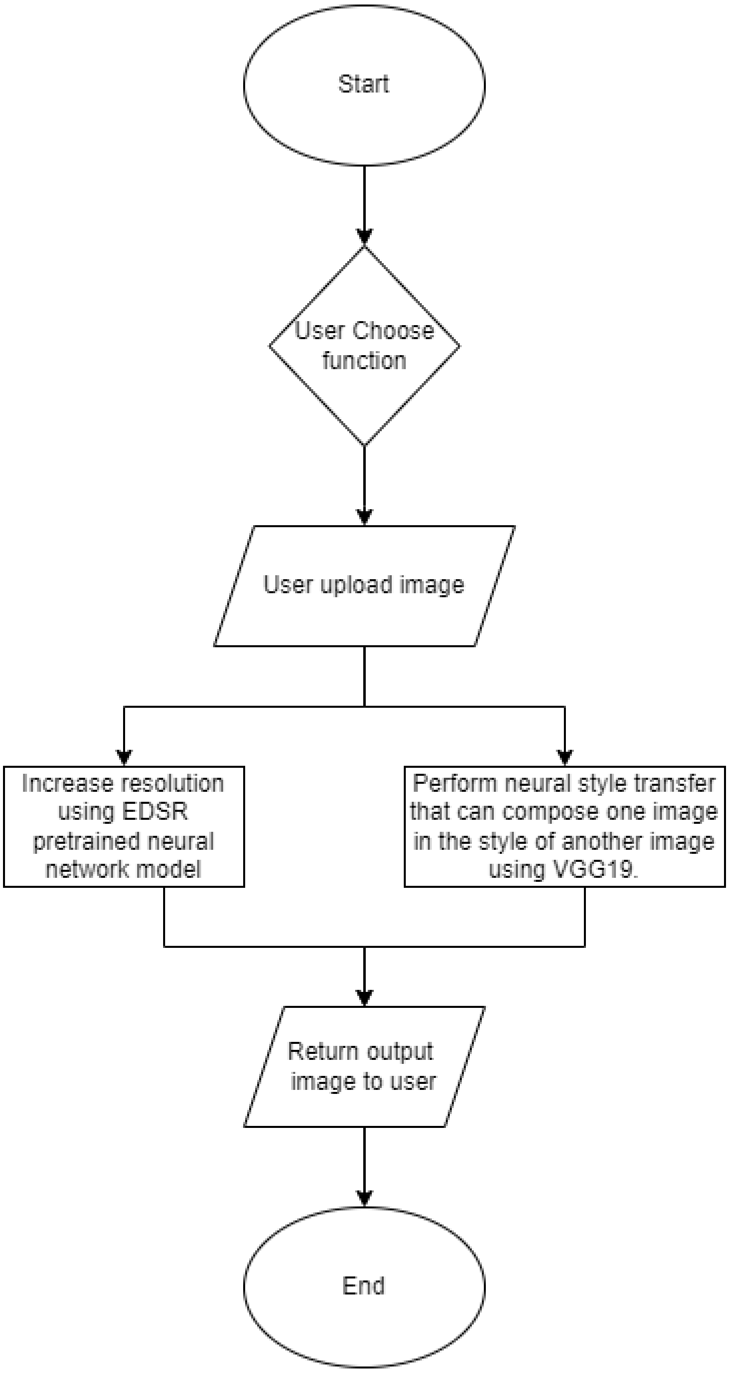

Start
User Choose function
User upload image
Increase resolution using EDSR pretrained neural network model
Perform neural style transfer that can compose one image in the style of another image using VGG19.
Return output image to user
End

Figure 4.4: Flowchart of ImageGenie

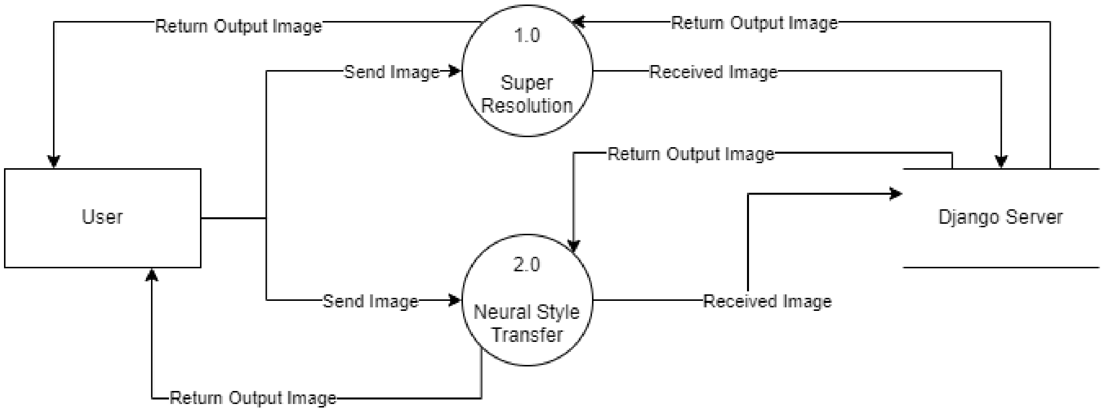


Figure 4.5: Data flow diagram of the system

Figure 4.4 and Figure 4.5 show the flowchart and data flow diagram of the system respectively. The user needs to upload an input image for the task super resolution and neural style transfer.

4.4 Conclusion

In this chapter, we presented the design phase of our project. We outlined the high-level system architecture, user interface design, technical design, and output design. We also provided detailed software design, considering different methodologies and approaches. The next chapter will focus on the implementation phase, where we will translate our design into a working system.

CHAPTER 5: IMPLEMENTATION

5.1 Introduction

For the implementation phase, Visual Studio Code was selected to develop the Django backend and also the Flutter frontend. Due to the integrated functionality of the plugin of Visual Studio Code, the process of developing the program can be made less complex. This chapter will go through the technical implementation of the system concerning the backend configuration, platform configuration, version control, software configuration and software development environment setup.

5.2 Software Development Environment Setup

Figure 5.1 shows the software development environment setup of this project. The component for the project is separated into two parts which are the frontend and backend.

Frontend is the user interface that the user is interact and it is the entry point for the user to use the function provided by the project. The frontend or user interface is developed using Flutter as depicted on Figure 5.2, which is an open source cross platform framework created by Google to build application.

On the other hand, the backend part is act as a server which handle the request from the user and perform related image enhancement functionality. The server is developed using Django framework as depicted on Figure 5.3 and the deep learning image enhancement is implemented using TensorFlow as depicted on Figure 5.4. Figure 5.5 depicts the OpenCV which is also used in the backend to carry out image pre-processing task.

The frontend communicate with the backend through REST-API which allow data to be transferred and exchanged between server and the mobile application. Figure 5.6 depicts the Visual Studio Code which is used as the Integrated Development Environment (IDE) in this project for both the server and the mobile application. The programming languages used for the frontend is Dart version 2.17.6 and Python version 3.10.4 as depicted on Figure 5.7 for the backend.

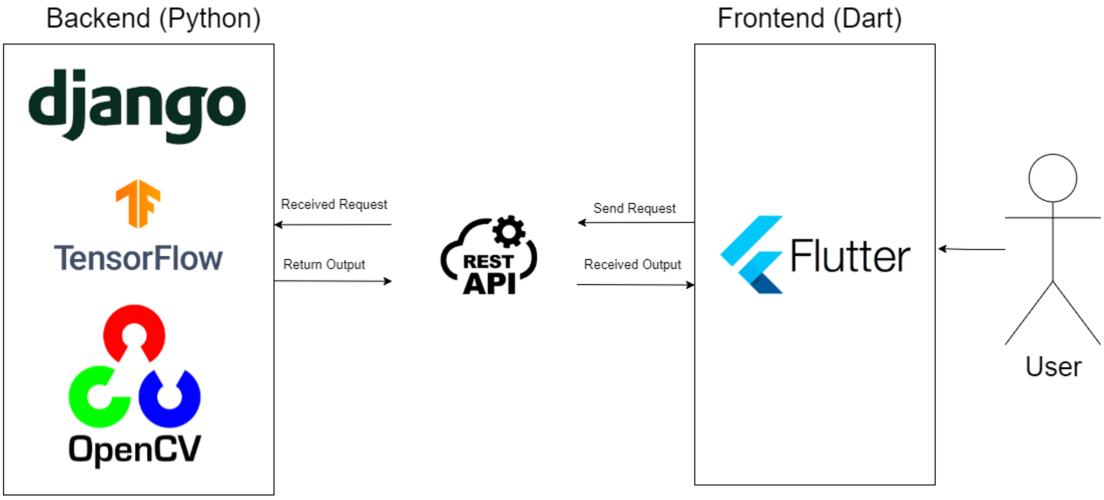


Figure 5.1: Overall Software Development Environment Setup

Figure 5.2: Flutter

Figure 5.3: Django

Figure 5.4: TensorFlow

Figure 5.5: OpenCV

Figure 5.6: Visual Studio Code

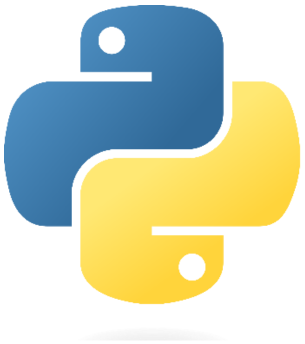


Figure 5.7: Python v3.10.4

Table 5.1: Overview of the Softwate Develoment Environment

Tools / Library	**Description**
Flutter	Open source cross platform framework to build application.
Django	Used to develop the server.
TensorFlow	Library to develop deep learning API.
OpenCV	Carry out image pre-processing task.
Python v3.10.4	Programming language used in Django, TensorFlow and OpenCV.
Visual Studio Code	Integrated Development Environment (IDE) in this project for both the server and the mobile application

5.3 Software Configuration Management

5.3.1 Configuration Environment Setup

This section will explain the environment configuration for both the backend and the frontend of the project.

5.3.1.1 Backend Configuration Environment Setup

The configuration environment setup for the backend environment was structure to ensure the smooth development and deployment of the project. To construct the backend, the user need to install Python version 3.10.4 from the official website. A configuration file was created to define the version of the Python library that used in this project. The configuration file is within the backend folder “django_api” and called “requirements.txt”. Inside the “requirements.txt” file, a list of the necessary libraries along with their specific versions was provided. For example, libraries such as Django for web framework, TensorFlow for deep learning, OpenCV for image processing, and Django Rest Framework for creating APIs were specified. By listing these packages along with their versions, the environment setup was reproducible, and any potential compatibility issues were minimized. This files can be used to install all the required library in order to run the backend.

The steps to construct the backend environment are listed below:

1) Firstly, the user need to download the source code of this project from GitHub, then the user need to navigate to the backend directory which is the “django_api” directory.

2) Then, type the following command in the command prompt to install relevant dependencies and libraries “pip install -r requirements.txt”

3) To ensure that the libraries were installed successfully, the user can run the following command to see a list of installed packages, “pip list”.

4) With the required libraries installed, the user can start running the backend server. The user can start the development server using the following command, “python manage.py runserver 0.0.0.0:8000”.

5.3.1.2 Frontend Configuration Environment Setup

In order to construct the frontend, the user need to install Flutter version 3.0.5 from the official website. This entailed downloading the SDK from the official Flutter website and subsequently extracting it to a chosen directory within the local machine's file system.

The steps to construct the frontend environment are listed below:

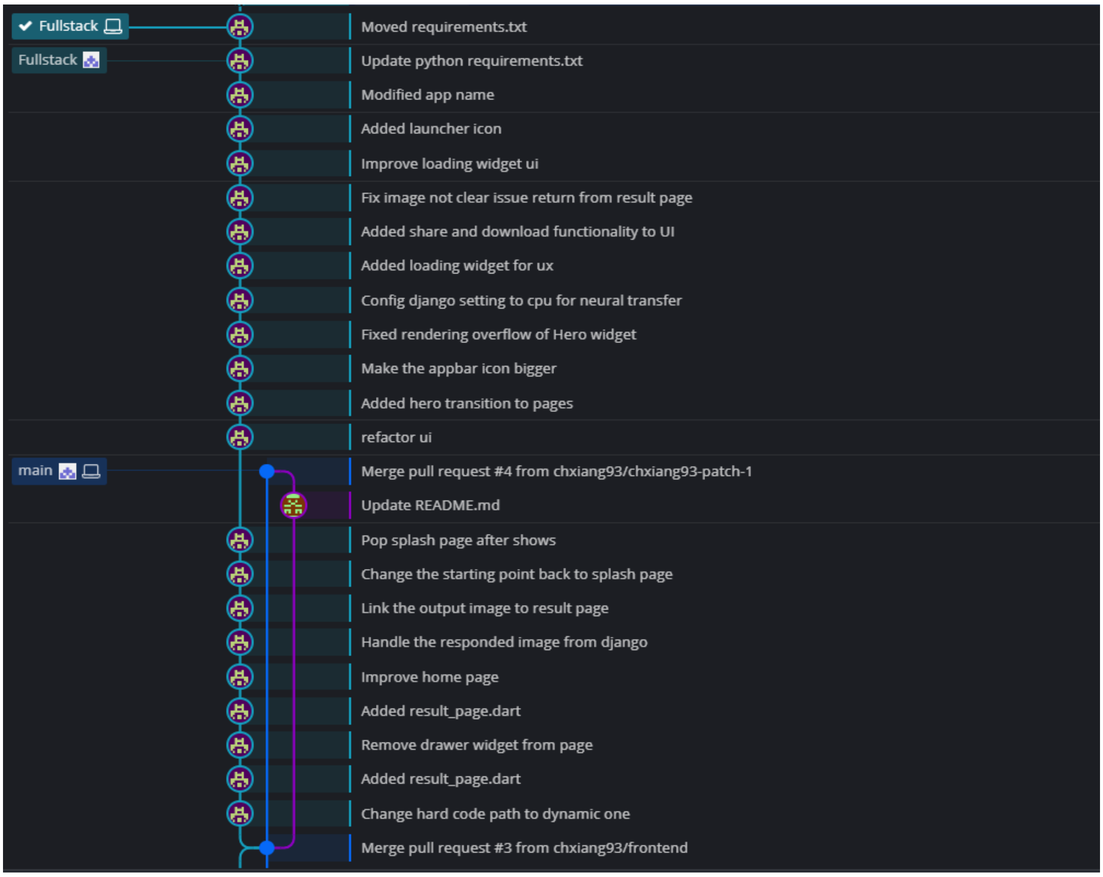
1) Firstly, the user need to download the source code of this project from GitHub, then the user need to navigate to the backend directory which is the “image_genie_app” directory.

2) Then, the user can start running the frontend application. The user can start the mobile application using the following command in the command prompt, “flutter run”.

5.3.2 Version Control Procedure

To ensure effective version control of the source code of the project, the Git version control system is employed. The project's source code and related files are stored in a central repository hosted on GitHub. A version control procedure must be implemented in the project to allow consistency of the system.

Version control procedures involve creating branches for specific features or fixes, making changes within these branches, and then merging them back into the main codebase. Regular commits, pull requests, and code reviews are essential practices to maintain code quality and consistency. By adhering to these version control procedures, developer can easily track changes and roll back to previous versions if needed. Figure 5.8 shows the development branches and commits made using Git version control system.



Moved requirements.txt
Update python requirements.txt
Modified app name
Added launcher icon
Improve loading widget ui
Fix image not clear issue return from result page
Added share and download functionality to UI
Added loading widget for ux
Config django setting to cpu for neural transfer
Fixed rendering overflow of Hero widget
Make the appbar icon bigger
Added hero transition to pages
refactor ui
Merge pull request #4 from chxiang93/chxiang93-patch-1
Update README.md
Pop splash page after shows
Change the starting point back to splash page
Link the output image to result page
Handle the responded image from django
Improve home page
Added result_page.dart
Remove drawer widget from page
Added result_page.dart
Change hard code path to dynamic one
Merge pull request #3 from chxiang93/frontend

Figure 5.8: Commit List Screenshot In GitKraken

5.4 Implementation Status

The development progress of each component/module as as follows. Table 5.2 shown the implementation status that involve the module name, description, duration to complete and date completed.

Table 5.2: Detail Implementation Status

Modules	**Description**	**Duration to Complete (days)**	**Date Completed**
Deep Learning Implementation	Implementation of deep convolutional neural network models for image enhancement task.	30	6 April 2023

Backend Development	Creation of the server to handle user requests and manage image enhancement functionality.	60	5 June 2023
Frontend Development	Design and development of the user interface for the mobile application.	30	27 August 2023

5.5 Conclusion

In this chapter, the software configuration environment management approach, version control procedures, and the implementation status of various modules have been discussed. The carefully designed software development environment ensures efficient development, version tracking, and successful integration of both frontend and backend functionalities.

CHAPTER 6: TESTING

6.1 Introduction

This chapter will go through the tests and steps to perform tests. The models for super resolution and neural style transfer will be compared and the best one will be chosen to implement in the mobile application.

6.2 Test Plan

6.2.1 Test Organization

In this project, different roles of the people are involved in the system for different responsibilities of the testing phase. These individuals are the core roles involved in this project. Table 6.1 shows the person who involved in the testing process.

Table 6.1: Roles and Responsibilities of Individuals Involved in the Testing Phase

Roles	**Responsibility**
Developer	The person who responsible to the development of the system which involve writing code and performing unit testing.
User	User of the system which responsible for performing beta tests to ensure the system is user friendly.
Dr Fauziah binti Kasmin	Supervisor of the Final Year Project.

6.2.2 Test Environment

In the test environment used for the testing, we will test the system with a mobile emulator virtual device with android API 30 and also android physical device which the user owned.

6.2.3 Test Schedule

Table 6.2 below shows the test schedules of the module based on the type of test conducted. In this part, the test schedule is a stage to ensure every testing run according to the plan.

Table 6.2: Test Schedule

Testing Activity	**Description**	**Remark**
Unit Test	To test every part of the system to ensure it is well functioned.	Done
Super Resolution Models Test	To test the performance of different super resolution models and identify the best one.	Done
Neural Style Transfer Models Test	To test the performance of different CNN models and identify the best one.	Done
UI/UX Test	To test is the user interface user friendly	Done

6.3 Test Strategy

The test strategies which applied in the testing of the system are white-box testing and black-box testing. Black-box testing is a way of system testing in which the testers do not know the technical or implementation aspect of the system while white-box testing is system testing done by the testers knows about the internal technical or code of the application and is done with the knowledge of implementation of the system. Hence in our case, black-box texting is targeted towards users while white-box texting is targeted towards developer.

6.3.1 Classes of Tests

6.3.1.1 Server Test

This test involves validating the functionality of the functions to identify whether it serves its intended requirement properly.

6.3.1.2 Super Resolution Models Test

This test aims to compute the performance of super resolution models EDSR, ESPCN and LapSRN using metrics PSNR and SSIM and the best one is identified and implemented in the mobile application.

6.3.1.3 Neural Style Transfer Models Test

This test aims to compute the performance of neural style transfer CNN models ResNet, MobileNet and VGG19 using metrics ArtFID and the best one is identified and implemented in the mobile application.

6.3.1.4 UI/UX Test

This test involves examing whether the graphical user interface serves its intended purposes.

6.4 Test Implementation

6.4.1 Experimental / Test Description

Tables below displays the actual functionality test case or experiment done during the testing process for all the modules. Table 6.3 below shows the test cases done to test the server which contain the image enhancement api to be consumed by the mobile application. Table 6.4 below displays the test cases done for the super resolution models. Table 6.5 below depicts the test cases done for the neural style transfer models. Table 6.6 below shows the test cases done for the UI/UX.

Table 6.3: Test Cases for Server Test

Category	Task / Description	Status
Functional	Ensure the deep learning API is working.	Functioning
Integration	Ensure that the connection can be established with mobile application via REST API.	Functioning
Navigation	Ensure that correct api is called with respective route or urls.	Functioning

Table 6.4: Test Cases for Super Resolution Models

Category	Task / Description	Status
Performance Evaluation	Evaluate the performance of super resolution models EDSR, ESPCN and LapSRN using metrics PSNR and SSIM. Choose the best one to implement in mobile application.	Complete

Table 6.5: Test Cases for Neural Style Transfer Models

Category	Task / Description	Status
Performance Evaluation	Evaluate the performance of neural style transfer by implementing different CNN models ResNet,	Complete

	MobileNet and VGG19 using metrics ArtFID. Choose the best one to implement in mobile application.	

Table 6.6: Test Cases for UI/UX

Category	**Task / Description**	**Status**
Functional	Ensure the mobile application can functions.	Functioning
Integration	Ensure the mobile application can establish connection to the server successfully.	Functioning
Navigation	Ensure the navigation between pages are smooth and correct page are navigated with the correct button.	Functioning

6.4.2 Test Data

The testing on the performance of super resolution models EDSR, ESPCN and LapSRN are using publicly avaiable benchmark dataset. The Set14 dataset is a dataset that consisting of 45 images commonly used for testing performance of image super resolution models.

On the other hand, the testing on the performance of neural style transfer CNN models ResNet, MobileNet and VGG19 use 10 content images with 3 style images which results in 30 testing images.

6.5 Test Results and Analysis

6.5.1 User Interface Result

The result of the user interface for the developed mobile application is shown below. The mobile application is created using Flutter framework.

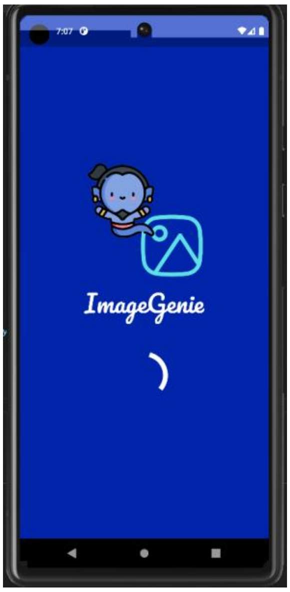


Figure 6.1: Splash Page

Figure 6.1 shows the splash page for ImageGenie which display a introduction page to user which consist of the icon of the application and animation of a circle running. After 2 seconds, the application will enter the home page.

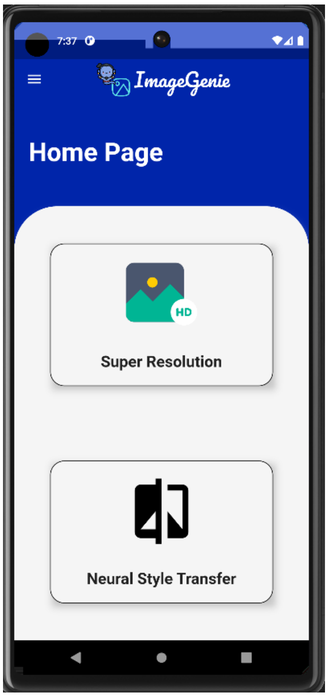


Figure 6.2: Home Page

Figure 6.2 shows the home page which consist of two card item which is clickable. The first card will navigate to the image super resolution page and the second card item will navigate to the neural style transfer page upon clicked.

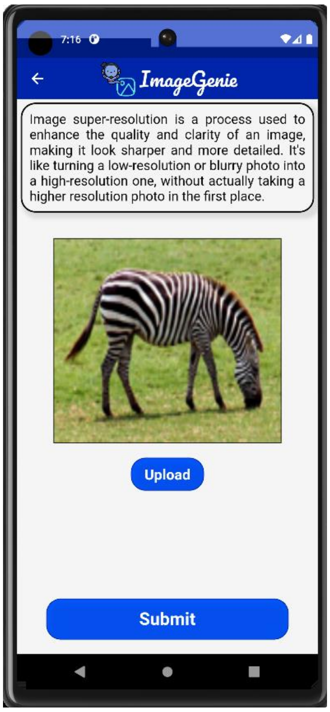
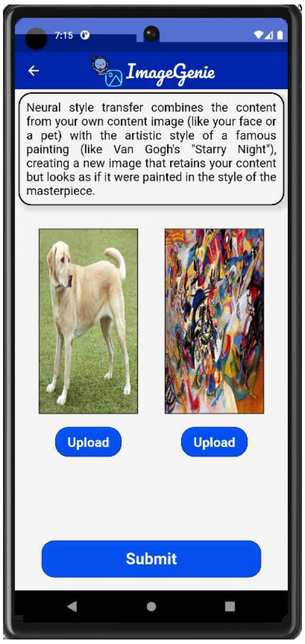


Image super-resolution is a process used to enhance the quality and clarity of an image, making it look sharper and more detailed. It's like turning a low-resolution or blurry photo into a high-resolution one, without actually taking a higher resolution photo in the first place.

Figure 6.3: Image Super Resolution Page

Figure 6.3 illustrates the image super resolution page which include the description of what is image super resolution, a upload button which can select an image from the device file system to carry out the image super resolution task. At the bottom of the page, a submit button is used to send the image to the Django server to process the image.



Neural style transfer combines the content from your own content image (like your face or a pet) with the artistic style of a famous painting (like Van Gogh's "Starry Night"), creating a new image that retains your content but looks as if it were painted in the style of the masterpiece.

Figure 6.4: Neural Style Transfer Page

Figure 6.4 displays the neural style transfer page which consist of the description of what is neural style transfer, two upload button to upload content image and style image respectively. At the bottom of the page, a submit button is used to send the images to the Django server to process the image.

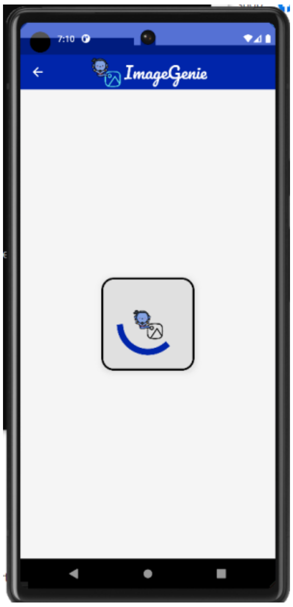


Figure 6.5: Loading Page After Submitted Input Image

Figure 6.5 shows the loading page for both the image super resolution and neural style transfer tasks. This page will indicate that the image is currently being processed at the server.

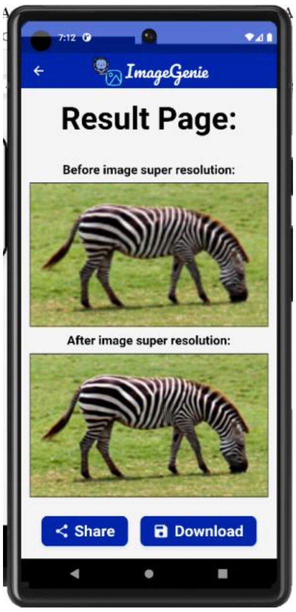


Figure 6.6: Result Page for Image Super Resolution

Figure 6.6 illustrates the result page for image super resolution task. The result shows the image before and after image super resolution. There is also share button which allow users to share the image to social media platform and a download button which can download the super resolved image to local device.

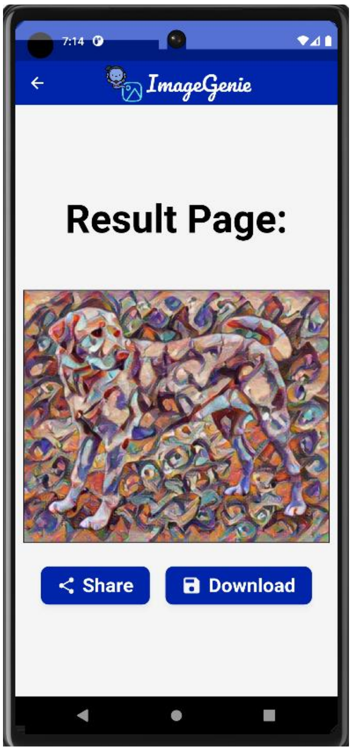


Figure 6.7: Result Page for Neural Style Transfer

Figure 6.7 displays the result page for neural style transfer task. There is also share button which allow users to share the image to social media platform and a download button which can download the super resolved image to local device.

6.5.2 Super Resolution Models Comparison Results

PSNR is a widely used metric to evaluate the quality of a reconstructed image. It measures the ratio between the maximum possible power of a signal (usually the original, unaltered image) and the power of the noise or distortion introduced by reconstruction. The PSNR is calculated using the mean squared error (MSE) between the original and reconstructed images and is typically expressed in decibels (dB). A higher PSNR value indicates a higher fidelity and lower distortion in the reconstructed image. The formula to calculate PSNR is shown below.

$$PSNR = 20log_{10}(\frac{\max_f}{\sqrt{MSE}})$$

Where the MSE (Mean Squared Error) is:

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}|f(i,j) - g(i,j)|^{2}$$

SSIM is a metric used to assess the similarity between two images. It is designed to capture both structural information and perceived changes in luminance, contrast, and structure. The final SSIM index value ranges between 0 and 1, with 1 indicating a perfect match and 0 indicating no similarity. Higher SSIM values correspond to images that are visually more similar. The formula to calculate SSIM is shown below.

$$SSIM(x,y) = \frac{2(\mu_x\mu_y + c1)(2\sigma_{xy} + c2)}{(\mu_x^2 + \mu_y^2 + c1)(\sigma_X^2 + \sigma_y^2 + c2)}$$

Figure 6.8, Figure 6.9 and Figure 6.10 illustrate the examples of image before and after undergoing super resolution using different super resolution models which are EDSR, ESPCN and LapSRN with metrics PSNR and SSIM computed.

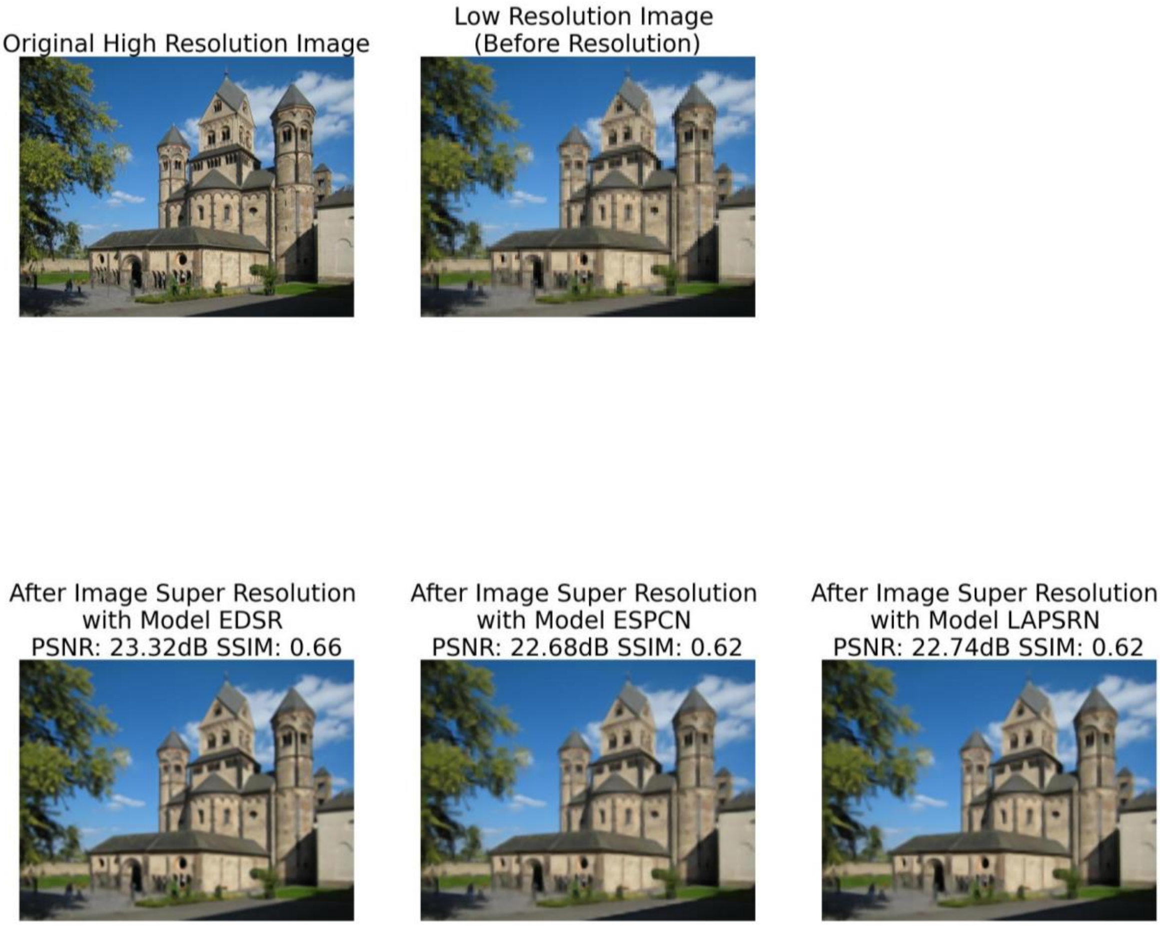

Original High Resolution Image
Low Resolution Image (Before Resolution)
After Image Super Resolution with Model EDSR PSNR: 23.32dB SSIM: 0.66
After Image Super Resolution with Model ESPCN PSNR: 22.68dB SSIM: 0.62
After Image Super Resolution with Model LAPSRN PSNR: 22.74dB SSIM: 0.62

Figure 6.8: Example of Image Before and After Undergoing Super Resolution Using Different Super Resolution Models

In the example of Figure 6.8, we can see that the EDSR can obtain a fairly clear image espcially on the edges of the castle. The edges is clear and solid compare to that of ESPCN and LapSRN. The color of the super resolved image of EDSR is also more vivid. The leaves in the super resolved image of EDSR is also reverted to match the original image very well.

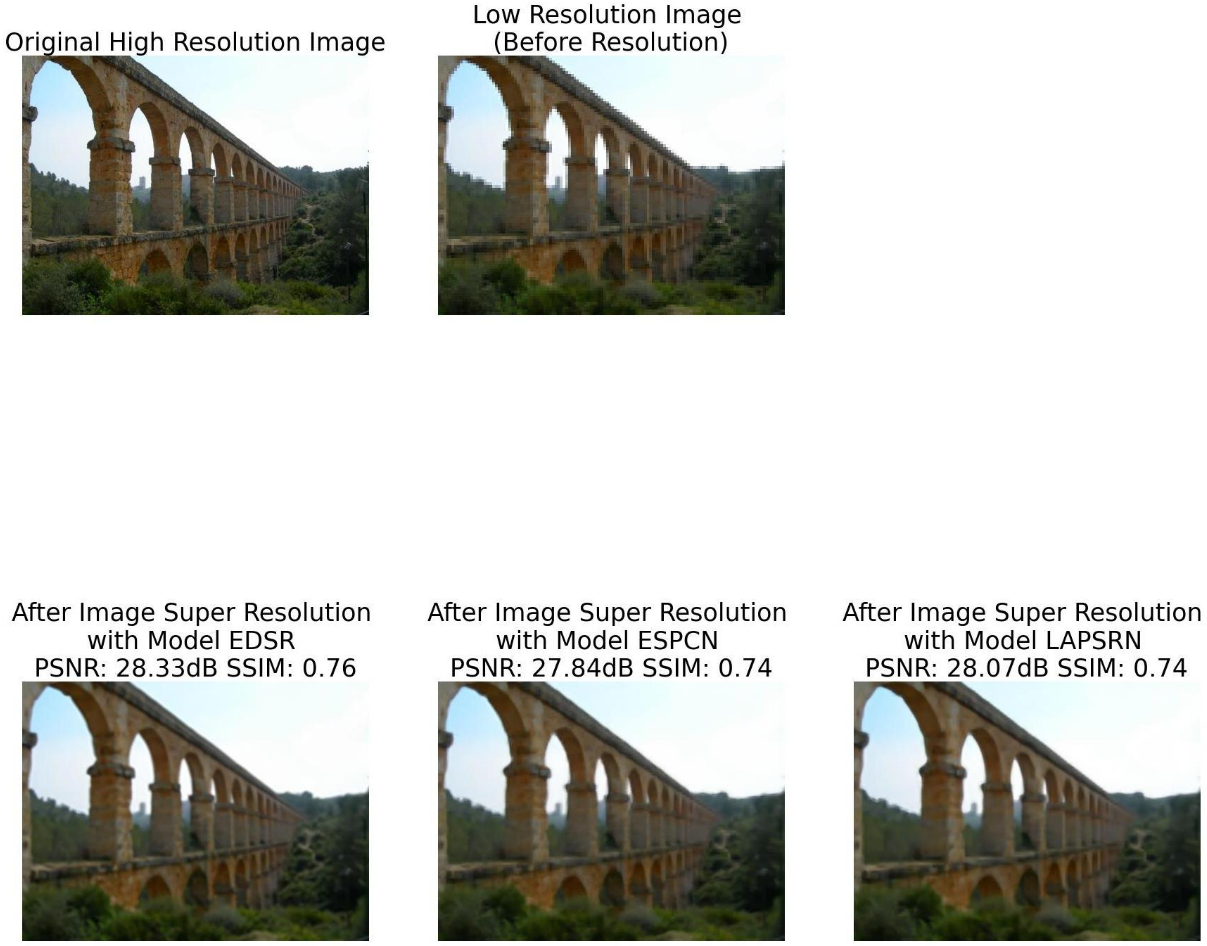

Original High Resolution Image
Low Resolution Image (Before Resolution)
After Image Super Resolution with Model EDSR PSNR: 28.33dB SSIM: 0.76
After Image Super Resolution with Model ESPCN PSNR: 27.84dB SSIM: 0.74
After Image Super Resolution with Model LAPSRN PSNR: 28.07dB SSIM: 0.74

Figure 6.9: Example of Image Before and After Undergoing Super Resolution Using Different Super Resolution Models

In the example of Figure 6.9, we can see that the EDSR can obtain a fairly clear image espcially on the edges of the bridge. The edges is clear and solid compare to that of ESPCN and LapSRN. The color of the super resolved image of EDSR is also more vivid.

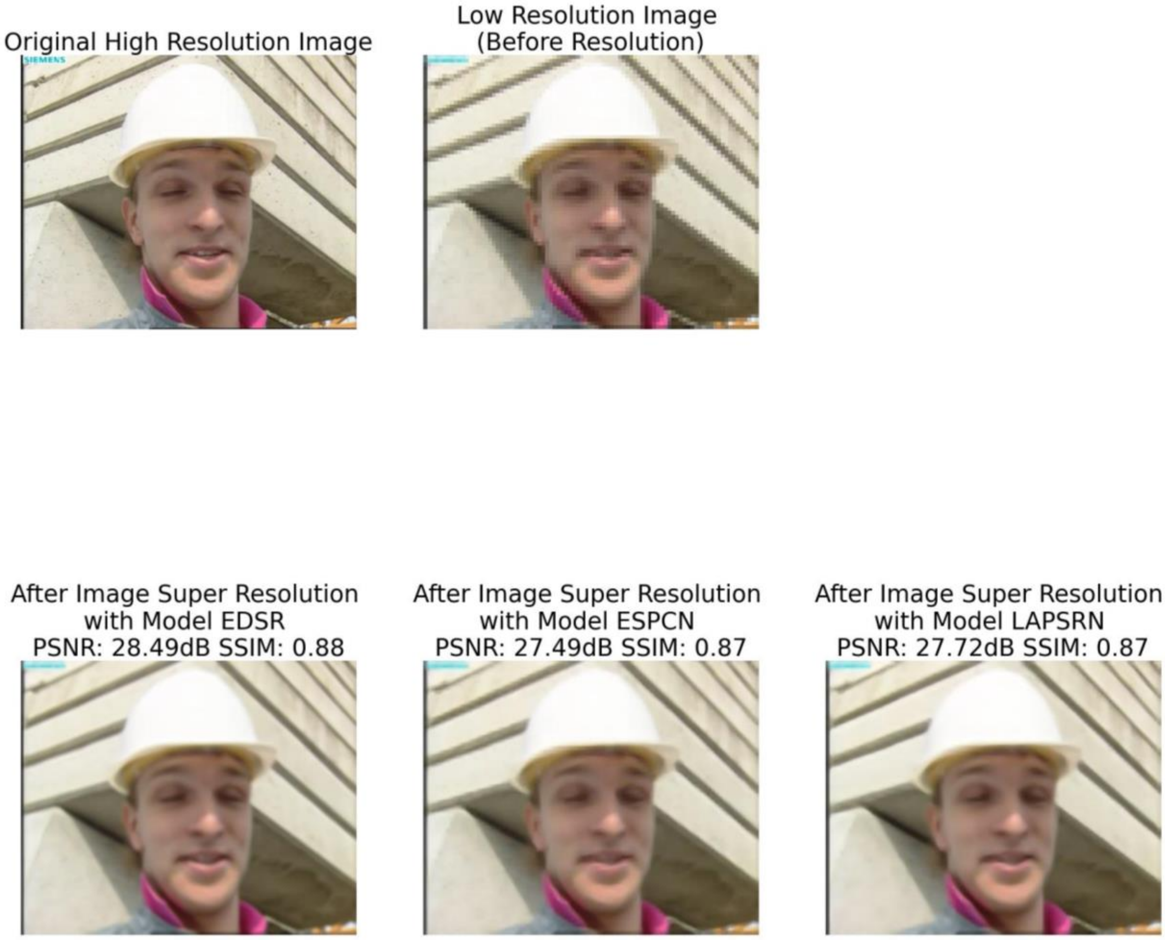

Original High Resolution Image
Low Resolution Image (Before Resolution)
After Image Super Resolution with Model EDSR PSNR: 28.49dB SSIM: 0.88
After Image Super Resolution with Model ESPCN PSNR: 27.49dB SSIM: 0.87
After Image Super Resolution with Model LAPSRN PSNR: 27.72dB SSIM: 0.87

Figure 6.10: Example of Image Before and After Undergoing Super Resolution Using Different Super Resolution Models

In the example of Figure 6.10, we can see that the EDSR can obtain a fairly clear image espcially on the edges of the face. The edges is clear and solid compare to that of ESPCN and LapSRN. Table 6.7 below shows the comparison of EDSR, ESPCN and LapSRN on 45 testing images in term of PSNR metric.

Table 6.7: Comparison of EDSR, ESPCN and LapSRN on 45 testing images in term of PSNR

Testing Images	**PSNR for EDSR**	**PSNR for ESPCN**	**PSNR for LapSRN**
0	21.878399	21.449944	21.461944
1	23.315562	22.679062	22.735566
2	26.111846	25.313282	25.338782

3	28.331049	27.844156	28.073571
4	25.161231	24.436621	24.416166
5	24.352942	23.829916	23.892548
6	25.604176	25.391552	25.375444
7	25.746619	24.826793	24.889295
8	22.416045	21.423278	21.528038
9	24.121323	23.227436	23.210929
10	31.673301	31.020633	31.050307
11	29.853931	29.072989	29.123638
12	25.282401	24.149216	24.236310
13	35.376153	34.610614	34.602020
14	28.493322	27.494960	27.717854
15	23.246038	22.808271	22.829830
16	30.852457	29.913209	30.079588
17	28.971673	28.023187	28.053250
18	26.674827	26.089015	26.254745
19	24.294739	23.505603	23.593175
20	28.999248	27.205111	27.424549
21	25.095656	24.648887	24.682025
22	29.440361	28.488648	28.656179
23	27.699100	26.900454	26.888237
24	23.151439	21.905960	21.911420
25	23.581601	22.954031	22.955849
26	25.139445	24.023265	24.242495
27	30.320529	29.606569	29.604108
28	27.090382	26.480125	26.561322
29	23.193817	22.485449	22.519594
30	26.380273	25.700184	25.773560
31	23.599615	22.969184	23.033792
32	26.221630	25.722356	25.738186
33	19.742097	19.208276	19.265547
34	24.712881	24.069620	24.147578
35	22.491323	21.618717	21.774899

36	25.345032	24.810082	24.929146
37	23.374532	22.680546	22.778980
38	24.352436	23.756972	23.864585
39	29.037429	28.584109	28.596353
40	29.448629	28.780615	28.979260
41	24.223719	23.476029	23.534227
42	28.565813	28.091029	28.066656
43	19.889328	19.301676	19.382013
44	19.070460	18.629056	18.658960

Table 6.8 below shows the comparison of EDSR, ESPCN and LapSRN on 45 testing images in term of SSIM metric.

Table 6.8: Comparison of EDSR, ESPCN and LapSRN on 45 testing images in term of SSIM

Testing Images	**PSNR for EDSR**	**PSNR for ESPCN**	**PSNR for LapSRN**
0	0.472662	0.433278	0.431201
1	0.656359	0.620835	0.623640
2	0.758264	0.722497	0.722390
3	0.755064	0.738041	0.739238
4	0.644504	0.598876	0.596753
5	0.662501	0.637258	0.639125
6	0.572598	0.563236	0.560277
7	0.767148	0.737864	0.736743
8	0.632921	0.579156	0.584296
9	0.634278	0.602325	0.599813
10	0.782253	0.767527	0.766945
11	0.797340	0.782878	0.780250
12	0.745024	0.698185	0.699202
13	0.906031	0.896399	0.894557
14	0.882911	0.871294	0.872565

15	0.470871	0.439503	0.438019
16	0.842549	0.820028	0.821825
17	0.775000	0.749686	0.746631
18	0.750239	0.719723	0.721052
19	0.685578	0.653757	0.655652
20	0.925153	0.898761	0.902238
21	0.524979	0.498562	0.498662
22	0.867426	0.847479	0.847000
23	0.771575	0.746298	0.744603
24	0.887611	0.851408	0.851200
25	0.637554	0.608960	0.607616
26	0.735710	0.689339	0.684886
27	0.791200	0.768186	0.767234
28	0.776591	0.755821	0.756196
29	0.646438	0.600733	0.601888
30	0.682612	0.655961	0.656564
31	0.583133	0.543723	0.545462
32	0.617482	0.592250	0.589678
33	0.551272	0.506636	0.511104
34	0.688460	0.657782	0.660761
35	0.728327	0.691758	0.696020
36	0.634119	0.613759	0.615675
37	0.647141	0.609682	0.615518
38	0.652878	0.624623	0.626986
39	0.744774	0.726512	0.724134
40	0.822835	0.808772	0.810451
41	0.690474	0.655303	0.655718
42	0.733318	0.719072	0.715839
43	0.450788	0.402676	0.407999
44	0.430602	0.382027	0.386180

Figure 6.11 below shows the metric PSNR of EDSR, ESPCN and LapSRN to the testing images. Figure 6.12 below shows the metric SSIM of EDSR, ESPCN and LapSRN to the testing images. The results are tested on 45 low resolution images which each image will be super resolved using EDSR, ESPCN and LapSRN respectively and the results are compared. The average PSNR for EDSR is 25.82dB, 25.09dB for ESPCN and 25.16dB for LapSRN. The average SSIM for EDSR is 0.70, 0.66 for ESPCN and 0.66 for LapSRN. EDSR obtains the best results in both of the metrics with 25.82dB in PSNR and 0.70 in SSIM. Table 6.9 below shown the average PSNR and SSIM for each super resolution models tested on 45 images.

Table 6.9: Average PSNR and SSIM for Super Resolution Models

Super Resolution Models	**Average PSNR (dB)**	**Average SSIM**
EDSR	25.82	0.70
ESPCN	25.09	0.66
LapSRN	25.16	0.66

Based on Figure 6.11 and Figure 6.12, EDSR achieved the highest performance in terms of PSNR and SSIM. This means that EDSR can improve the quality of the image with the best results. Hence, we will utilize EDSR models since it provides the best result for image super resolution.

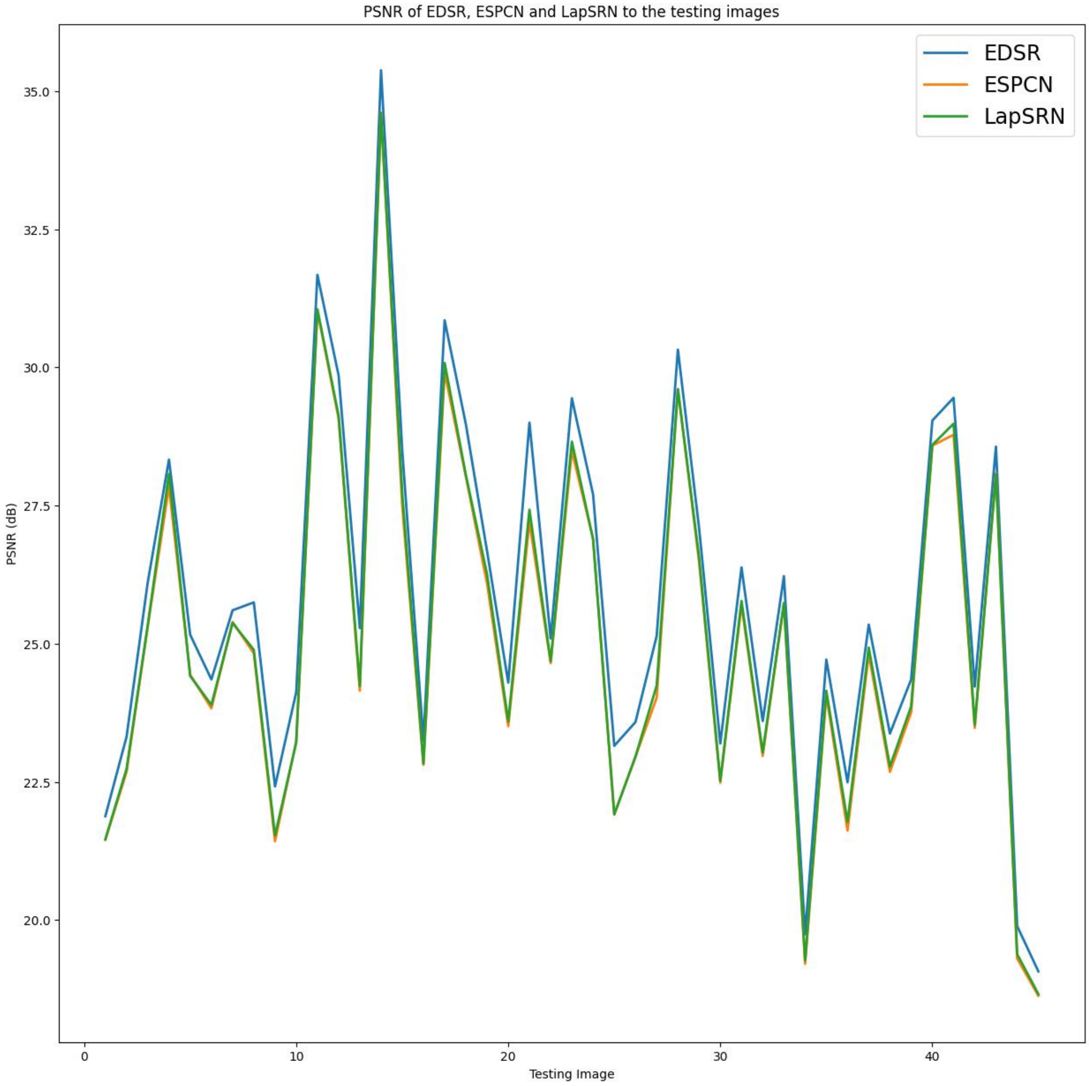


Figure 6.11: PSNR of EDSR, ESPCN and LapSRN to the testing images

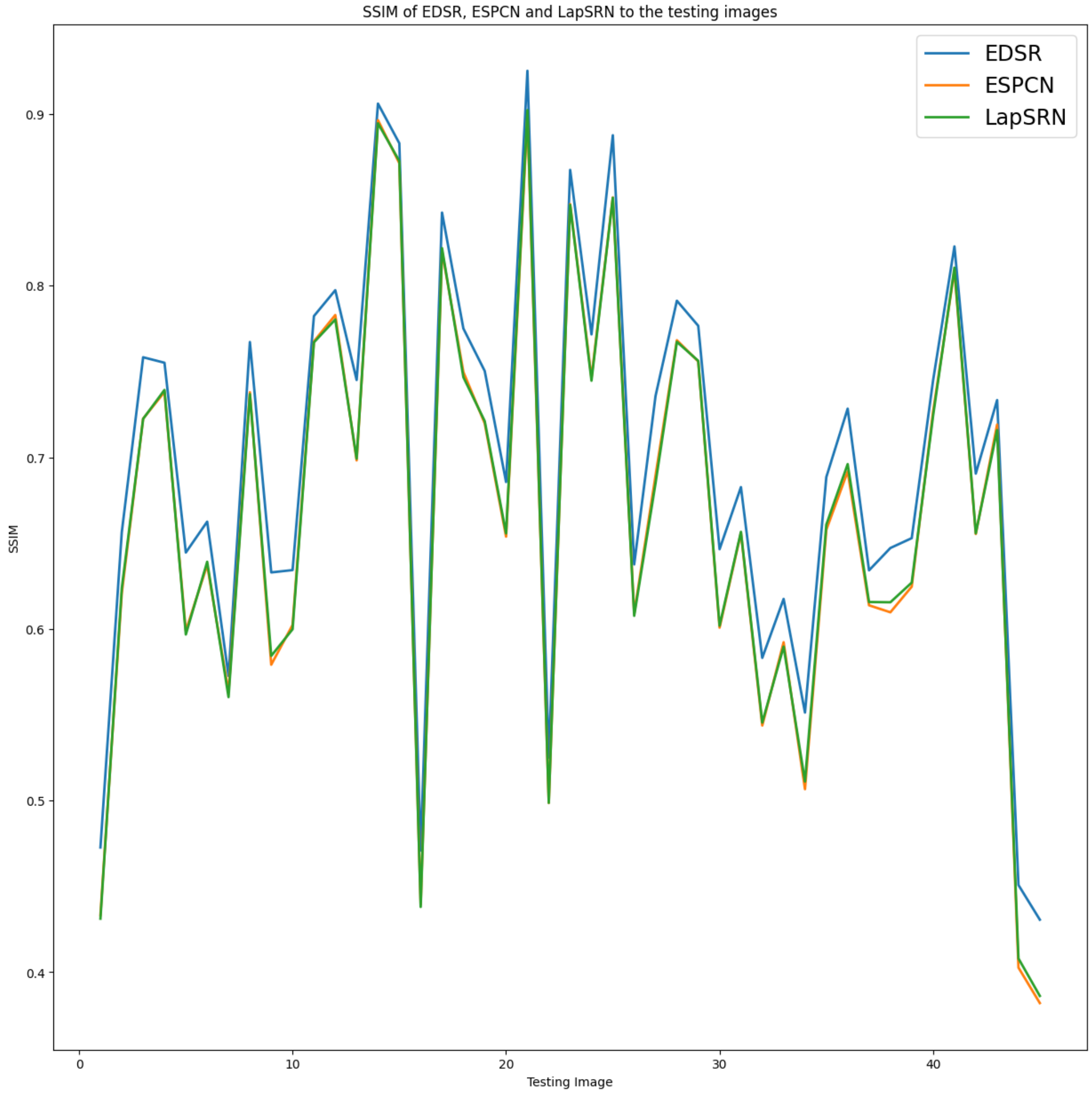


Figure 6.12: SSIM of EDSR, ESPCN and LapSRN to the testing images

6.5.3 Neural Style Transfer CNN Models Comparison Results

The test strategy in the neural style transfer is that different CNN models which are ResNet, MobileNet and VGG19 were employed and the performance or quality was measured using evaluation metric ArtFID. The ArtFID metric is used for assessing the quality of neural style transfer technique and is inspired by the Fréchet Inception Distance (FID) that is used to evaluate the quality of generated images. ArtFID measures the perceptual similarity between the stylized image and a reference image, capturing both the content and style aspects. The authors conduct extensive experiments using various neural style transfer algorithms and datasets that are labeled by artists to demonstrate the effectiveness of the ArtFID metric (Wright & Ommer, 2022). They compare ArtFID against other metrics, showcasing its ability to align better with human perception of style transfer quality. Equation below depicts the formula to calculate ArtFID (Wright & Ommer, 2022).

$$ArtFID\left(X_g, X_c, X_s\right) = (1 + \frac{1}{N}\sum_{i=1}^{N} d\left(X_c^{(i)}, X_g^{(i)}\right)) \cdot (1 + FID\left(X_s, X_g\right))$$

Each CNN models powered neural style transfer technique will experiment in processing 10 content images with 3 style images which will result in 30 output image which combine the content images with the style of style images. The 30 images will then be evaluated using ArtFID metric with higher values indicating better performance.

Style Image 1

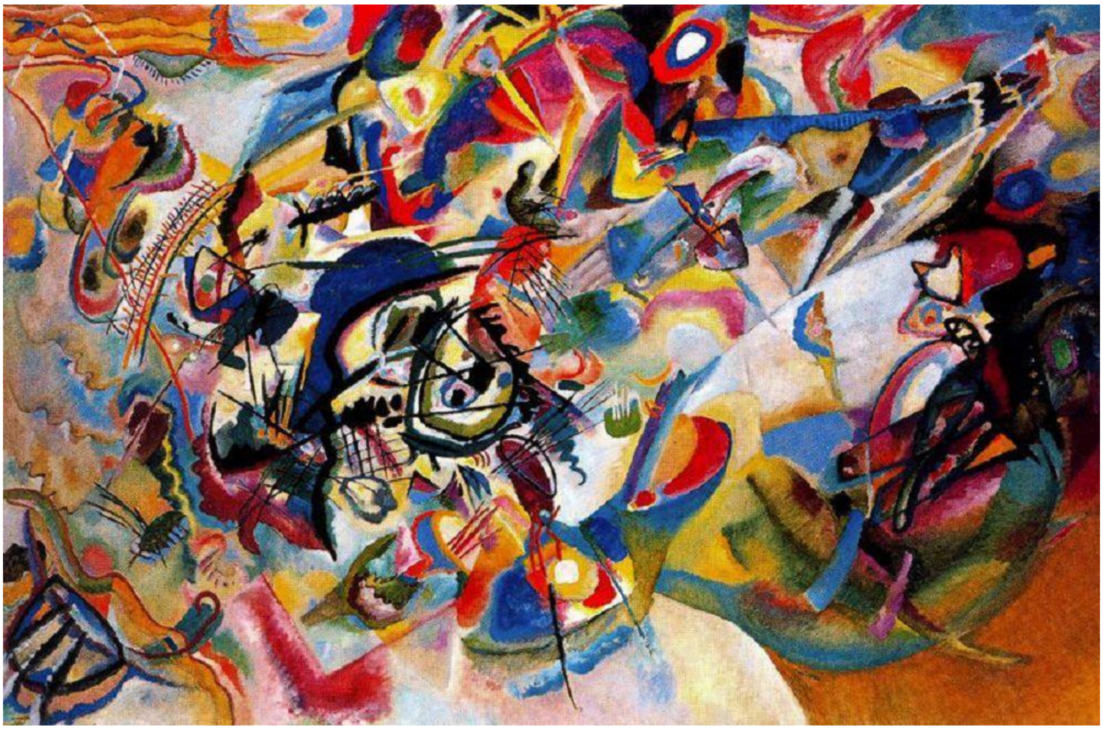


Figure 6.13: Style Image 1

Figure 6.13 above shows one of the style image used in evaluating the quality of the neural style transfer technique. Table 6.10 below displays the ArtFID value computer for all the CNN models for style image 1 with 10 content images.

Table 6.10: ArtFID Value of CNN Models for Style Image 1

Testing Image	ArtFID for ResNet	ArtFID for MobileNet	ArtFID for VGG19
1	48.61	47.00	51.80
2	54.74	42.43	54.94
3	59.67	52.33	62.55
4	61.90	54.27	67.69
5	76.65	61.49	74.59
6	76.88	62.30	62.35
7	58.84	64.41	60.92
8	57.67	52.44	58.81
9	62.33	53.95	70.65
10	65.96	57.15	71.48

Table 6.11: ArtFID Value Properties for Style Image 1

Models	Mean	Variance	Standard Deviation
ResNet	62.32	71.69	8.46
MobileNet	54.77	42.54	6.52
VGG19	63.57	49.63	7.04

Table 6.11 shows the ArtFID value properties which include mean, variance and standard deviation. MobileNet has the lowest mean ArtFID value which is 54.77 among the three models, indicating that it, on average, performs worse in preserving the style from style image 1 and content of the original images during the neural style transfer process. VGG19 has the highest mean ArtFID value which is 63.57, suggesting that it perform better than the other models in terms of style preservation. ResNet has the highest standard deviation among the models, indicating that its ArtFID values are more spread out, possibly resulting in a wider range of performance outcomes. MobileNet has the lowest standard deviation, suggesting that its performance is more consistent across the evaluated images.

Hence, VGG19 appears to be the most effective model among the three models, as it has the highest mean ArtFID value in the neural style transfer technique done toward 10 content images and style image 1.

Style Image 2

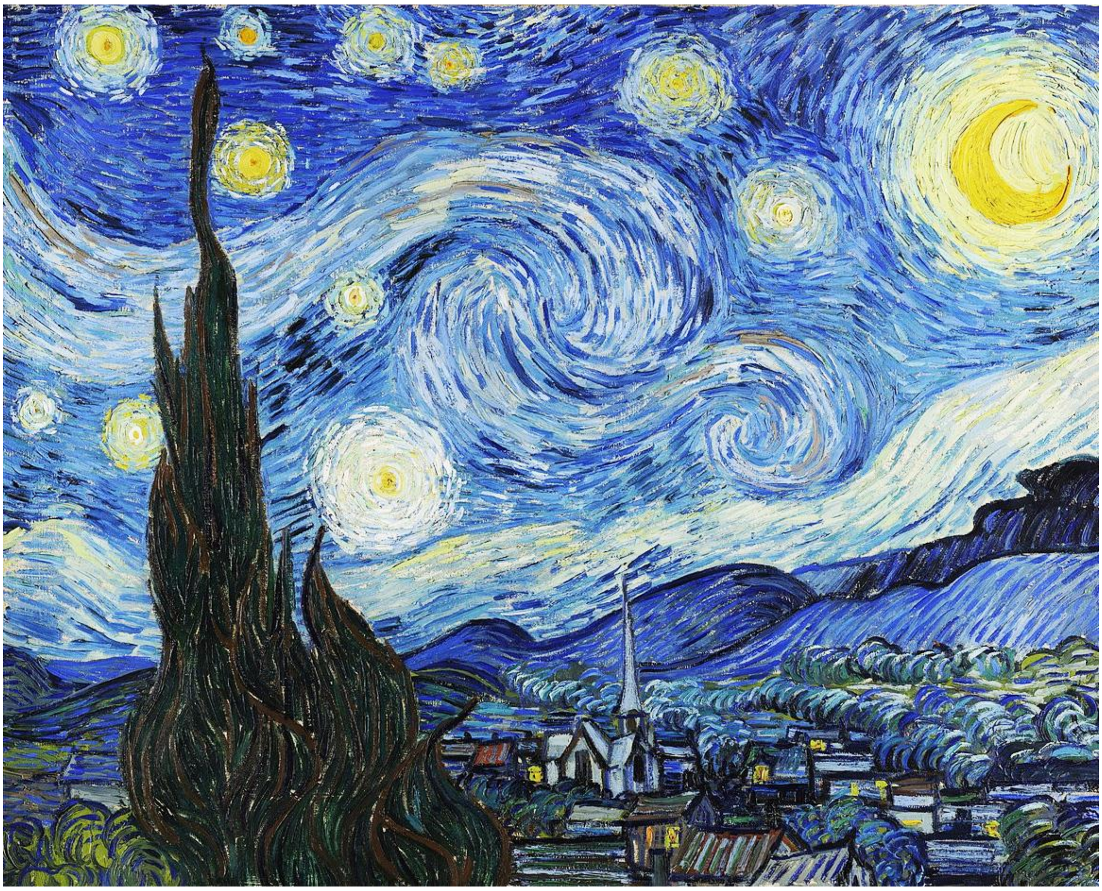


Figure 6.14: Style Image 2

Figure 6.14 above shows one of the style image used in evaluating the quality of the neural style transfer technique. Table 6.12 below displays the ArtFID value computer for all the CNN models for style image 2 with 10 content images.

Table 6.12: ArtFID Value of CNN Models for Style Image 2

Testing Image	**ArtFID for ResNet**	**ArtFID for MobileNet**	**ArtFID for VGG19**
11	29.74	27.27	37.83

12	30.74	37.66	39.22
13	26.70	37.20	40.55
14	34.48	37.59	44.94
15	34.85	49.55	49.33
16	29.73	45.25	34.33
17	33.68	35.02	48.95
18	35.50	38.44	44.76
19	43.69	50.53	50.67
20	33.85	37.69	49.34

Table 6.13: ArtFID Value Properties for Style Image 2

Models	**Mean**	**Variance**	**Standard Deviation**
ResNet	33.29	19.25	4.38
MobileNet	39.62	44.19	6.64
VGG19	43.99	29.37	5.41

Table 6.13 shows the ArtFID value properties which include mean, variance and standard deviation. ResNet has the lowest mean ArtFID value which is 33.29 among the three models, indicating that it, on average, performs worse in preserving the style from style image 2 and content of the original images during the neural style transfer process. VGG19 has the highest mean ArtFID value which is 43.99, suggesting that it perform better than the other models in terms of style preservation. MobileNet has the highest standard deviation among the models, indicating that its ArtFID values are more spread out, possibly resulting in a wider range of performance outcomes. ResNet has the lowest standard deviation, suggesting that its performance is more consistent across the evaluated images.

Hence, VGG19 appears to be the most effective model among the three models, as it has the highest mean ArtFID value in the neural style transfer technique done toward 10 content images and style image 2.

Style Image 3

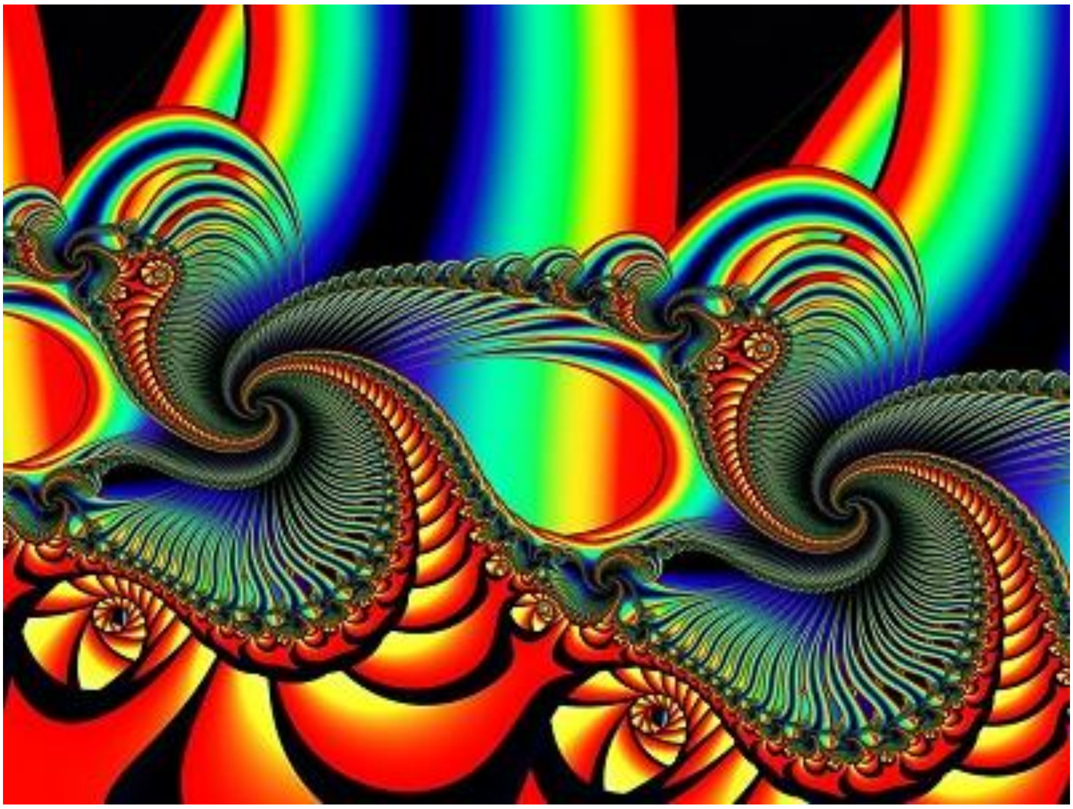


Figure 6.15: Style Image 3

Figure 6.15 above shows one of the style image used in evaluating the quality of the neural style transfer technique. Table 6.14 below displays the ArtFID value computer for all the CNN models for style image 3 with 10 content images.

Table 6.14: ArtFID Value of CNN Models for Style Image 3

Testing Image	ArtFID for ResNet	ArtFID for MobileNet	ArtFID for VGG19
21	31.02	22.23	25.30
22	37.55	30.50	30.12
23	39.10	31.91	31.08
24	31.39	32.95	39.86
25	41.99	27.99	35.32
26	46.50	30.60	29.79
27	30.37	28.47	36.22

28	32.86	27.38	29.45
29	28.70	36.71	28.61
30	34.07	29.60	33.63

Table 6.15: ArtFID Value Properties for Style Image 3

Models	**Mean**	**Variance**	**Standard Deviation**
ResNet	35.35	29.86	5.46
MobileNet	29.83	13.14	3.62
VGG19	31.93	16.53	4.06

Table 6.15 shows the ArtFID value properties which include mean, variance and standard deviation. MobileNet has the lowest mean ArtFID value which is 29.83 among the three models, indicating that it, on average, performs worse in preserving the style from style image 3 and content of the original images during the neural style transfer process. ResNet has the highest mean ArtFID value which is 35.35, suggesting that it perform better than the other models in terms of style preservation. ResNet has the highest standard deviation among the models, indicating that its ArtFID values are more spread out, possibly resulting in a wider range of performance outcomes. MobileNet has the lowest standard deviation, suggesting that its performance is more consistent across the evaluated images.

Hence, ResNet appears to be the most effective model among the three models, as it has the highest mean ArtFID value in the neural style transfer technique done toward 10 content images and style image 3. In my opinion, ResNet stand out in style image 3 because the colour combination for style image 3 is more contrast and complex than that of style image 1 and 2.

All Dataset

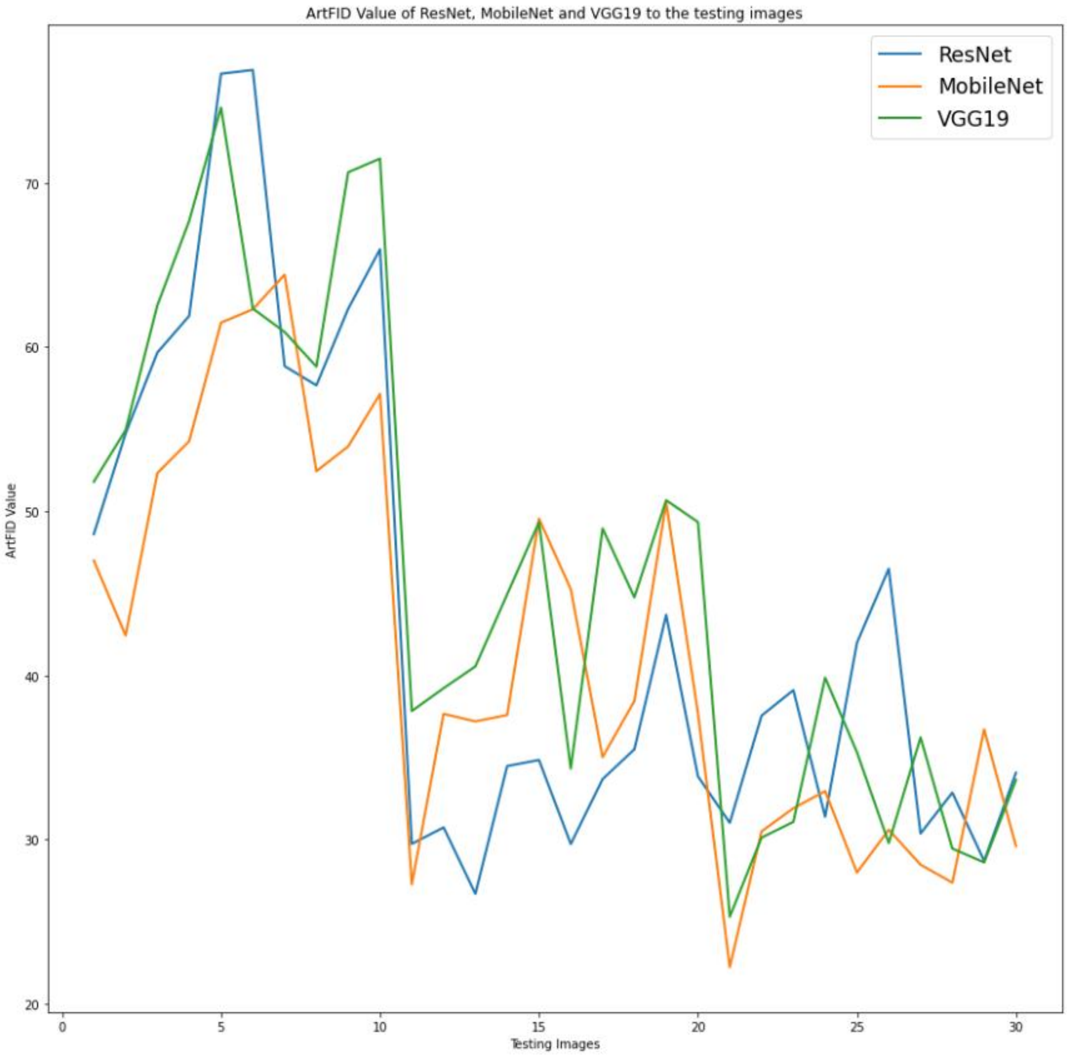


Figure 6.16: ArtFID Value for ResNet, MobileNet and VGG19 on Neural Style Transfer Technique Tested on 30 Images

Figure 6.16 above displays the graph of ArtFID value for ResNet, MobileNet and VGG19 on neural style transfer technique tested on 30 images.

Table 6.16: ArtFID Value Properties for all Dataset

Models	**Mean**	**Variance**	**Standard Deviation**
ResNet	42.10	40.26	6.10
MobileNet	39.96	33.29	5.59

VGG19	45.97	31.84	5.50

Table 6.16 shows the ArtFID value properties which include mean, variance and standard deviation for all three style images. MobileNet has the lowest mean ArtFID value which is 39.96 among the three models, indicating that it, on average, performs worse in preserving the style and content of the original images during the neural style transfer process. VGG19 has the highest mean ArtFID value which is 45.97, suggesting that it performs better than the other models in terms of style preservation. ResNet has the highest standard deviation among the models, indicating that its ArtFID values are more spread out, possibly resulting in a wider range of performance outcomes. VGG19 has the lowest standard deviation, suggesting that its performance is more consistent across the evaluated images.

In conclusion, VGG19 appears to be the most consistent and effective model among the three models, as it has the highest mean ArtFID value in the neural style transfer technique done toward 10 content images and three style images.

6.6 Conclusion

This chapter has outlined our comprehensive testing phase, which is crucial for ensuring the success of our project. The testing plan, strategy, and classifications were elaborated upon. We provided insights into test implementation, test data, and detailed records of test results and analysis. Moving forward, the subsequent chapter will draw conclusions from our testing phase and shed light on the upcoming activities in our project's development.

CHAPTER 7: PROJECT CONCLUSION

7.1 Observation on Weaknesses and Strengths

Throughout the development of this project, we observed both strengths and weaknesses in the development of the system, ImageGenie, an mobile application designed to enhance image resolution and perform neural style transfer.

First and foremost, we will discuss the project's strengths. The mobile application has a user friendly interface as it enables users to seamlessly engage with the image enhancement functions. Besides, this project provide a quick and easy way to enhance the quality of low-resolution image and perform neural style transfer.

However, this project also have weaknesses, including huge computing power is required to perform the image enhancement operations. Without an excellent GPU, the inference time of the image operation will be longer. Other than that, while ImageGenie generally delivered reliable results, there was still variability in image enhancement outcomes. This variability, primarily observed in the performance of VGG19 in neural style transfers, suggests room for improvement.

7.2 Propositions for Improvement

In this project, several propositions for improvement can be considered. One of the improvement that can be done is continuously refining and incorporating state-of-the-art deep learning models for image resolution enhancement can elevate the application's performance. Another improvement that can also be done is to further enhance the quality and consistency of neural style transfer, exploring advanced neural style transfer models beyond VGG19 that could yield improved artistic effects. Last but not least, we can also expanding the application's capabilities to include a broader scope of image processing techniques, such as object recognition, segmentation, and image denoising which would increase its versatility and utility.

7.3 Project Contribution

To sum up, this project can serve as a valuable reference for researchers and students interested in image processing and AI applications with the successful implementation and evaluation of various deep learning models for image enhancement, along with the development of a user-friendly application. ImageGenie make it a useful tool for individuals seeking to improve image quality and create captivating artistic images.

7.4 Conclusion

In conclusion, this project aimed to address the growing demand for image enhancement tools in the digital age. Through the integration of advanced deep learning models and a user-friendly interface, ImageGenie successfully achieved its objectives of enhancing image resolution and enabling neural style transfers. While the project demonstrated strengths in terms of effective image enhancement, consistency, and usability, there is room for improvement in reducing performance variability and expanding the application's feature set.

Overall, ImageGenie represents a promising step toward providing users with accessible and powerful image enhancement capabilities. As technology continues to advance, there are exciting opportunities to further refine and expand this application to meet evolving user needs and expectations in the field of image processing and artificial intelligence.

REFERENCES

1. Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). *A Neural Algorithm of Artistic Style*. http://arxiv.org/abs/1508.06576
2. Horé, A., & Ziou, D. (2013). Is there a relationship between peak-signal-to-noise ratio and structural similarity index measure? *IET Image Processing*, *7*(1), 12–24. https://doi.org/10.1049/iet-ipr.2012.0489
3. Johnson, J., Alahi, A., & Fei-Fei, L. (2016). *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. http://arxiv.org/abs/1603.08155
4. Lai, W.-S., Huang, J.-B., Ahuja, N., & Yang, M.-H. (2017). *Fast and Accurate Image Super-Resolution with Deep Laplacian Pyramid Networks*. http://arxiv.org/abs/1710.01992
5. Lim, B., Son, S., Kim, H., Nah, S., & Lee, K. M. (2017). *Enhanced Deep Residual Networks for Single Image Super-Resolution*. http://arxiv.org/abs/1707.02921
6. Ong Si Ci. (2023). *Image Super Resolution: A Comparison between Interpolation & Deep Learning-based Techniques to Improve Clarity of Low-Resolution Images*. https://medium.com/htx-s-s-coe/image-super-resolution-a-comparison-between-interpolation-deep-learning-based-techniques-to-25e7531ab207
7. Schulter, S., Leistner, C., & Bischof, H. (2015). *Fast and Accurate Image Upscaling with Super-Resolution Forests*. https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Schulter_Fast_and_Accurate_2015_CVPR_paper.pdf
8. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., & Wang, Z. (2016). *Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network*. http://arxiv.org/abs/1609.05158
9. Simonyan, K., & Zisserman, A. (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. http://arxiv.org/abs/1409.1556
10. Tashildar, A., Shah, N., Gala, R., Giri, T., & Chavhan, P. (2020). APPLICATION DEVELOPMENT USING FLUTTER. In *International Research Journal of Modernization in Engineering Technology and Science @International Research Journal of Modernization in Engineering*. https://www.irjmets.com/uploadedfiles/paper/volume2/issue_8_august_2020/3180/1628083124.pdf

11. Wright, M., & Ommer, B. (2022). *ArtFID: Quantitative Evaluation of Neural Style Transfer*. http://arxiv.org/abs/2207.12280
12. Zeyde, R., Elad, M., & Protter, M. (2012). On Single Image Scale-Up Using Sparse-Representations. In J.-D. Boissonnat, P. Chenin, A. Cohen, C. Gout, T. Lyche, M.-L. Mazure, & L. Schumaker (Eds.), *Curves and Surfaces* (pp. 711–730). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-27413-8_47
13. Malaysian Communications and Multimedia Commission (MCMC). (2021). Hand Phone Users Survey. Retrieved from https://www.mcmc.gov.my/skmmgovmy/media/General/pdf2/FULL-REPORT-HPUS-2021.pdf
14. Ipsos Malaysia. (2020). Digital Trends Survey. Retrieved from https://www.ipsos.com/en-my
15. Ipsos Malaysia. (2021). Digital 2021: Malaysia. Retrieved from https://datareportal.com/reports/digital-2021-malaysia