Face sketch colorization using conditional normalizing flow

by

Ruiqin Pi

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science
University of Alberta

Abstract

To colorized a face sketch involves adding appropriate color to the original sketch while maintaining the sketch's structure. This task combines the concept of gray-scale image colorization, which focuses on introducing color elements without changing the image's geometry, and image translation, which adds missing luminance and chrominance information to transform a sketch to a color image.

However, existing works for gray-scale image colorization tend to produce satisfactory colorization results only when they are provided with a rich and detailed gray-scale image as input. In contrast, sketches have much less information because of its minimalistic nature and the absence of shading or texture details, which makes the colorization process more challenging, as it requires the model to not only add the necessary color elements but also infer the missing details based on the sketch's structure. Some researchers tend to address this problem using image translation techniques. State-of-the-art approaches using Generative Adversarial Networks (GANs) based models are preferred due to their impressive performance and realistic results. Nevertheless, the semantic difference between synthesized images and their corresponding input sketches remains a key challenge that needs further investigation. Recently, normalizing flow, which is a new generative model, with ability to losslessly encode and decode data, has shown potential to tackle the face sketch colorization problem.

In this thesis, we first focus on losslessly converting sketches and their corresponding color images to high-dimensional latent features using a conditional normalizing flow within an Encoder-Decoder architecture. This framework adopts an inverse learning mechanism to simultaneously learn the color encoding and decoding processes. To ensure preserving geometry of the sketch while maintaining high-quality colorization, we integrate a Feature Aggregation module into the coupling block of our normalizing flow. This module can adaptively fuse sketch and color information within a high-dimensional space. Moreover, our normalizing flow is designed to be memory-efficient, which requires much less computational resources compared to that of similar models. We also demonstrate the effectiveness of our method to preserve the input geometry and to achieve perceptually satisfying colorization quality using both qualitative and quantitative evaluations comparisons with existing methods. This research contributes to advancing the field of face sketch colorization, and providing an innovative solution that addresses some of the limitations of existing approaches.

Acknowledgements

First and foremost, I would like to express my profound gratitude to my professor, Dr. Herb Yang, for his unwavering support and guidance throughout the course of my research. Through numerous engaging discussions, he not only inspired me to choose a research topic that closely aligned with my interests but also consistently offered insightful suggestions during our weekly meetings. His invaluable input, especially during times when I faced challenges, helped me overcome obstacles and find new directions. Without his assistance, completing my work would have been significantly more difficult. Furthermore, I am deeply appreciative of the access he granted me to the school's computer servers, which greatly facilitated the training of my models and execution of my experiments.

Additionally, I would like to extend my heartfelt thanks to my teammates in our research group, who have consistently shown enthusiasm in addressing my questions and have generously shared resources, such as GPU usage, when needed. I am also deeply grateful for the support from Broderick Wood, the server manager of the SciTech team within the Information Services and Technology (IST) group. Despite his busy schedule, he made it a priority to help me access more computational resources during the most challenging periods of my research, ensuring that I could quickly overcome obstacles and continue making progress.

Lastly, I would like to express my profound gratitude to the University of Alberta for providing me with the opportunity to pursue research in my areas of interest, allowing me to develop as a teaching assistant, and funding my graduate studies. I am forever indebted to my parents, whose unwavering and unconditional support has been a constant source of strength throughout my academic journey.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Looking back in history, creativity and artistic imagination have always been integrated into human drawings and paintings, which also act as a means of preserving civilizations and of cultural exchange. Among these works, the focus often lies on individuals or matters related to humans. However, in the early days of human history, due to the limitations of colorization techniques, artists had to rely on their skills for color sketches to add vibrancy and life to sketches or engravings. This labor-intensive and time-consuming process necessitated a profound understanding of the importance of color composition and its use in design.

Besides, with the advent of photography in the 19th century, a large number of photographs of people and pictures of scenery and buildings have emerged, the precious historical images have demonstrated the appearance of humanity during that period of history. However these photographs and images are still in black and white, leaving a significant gap in our understanding of the true appearance of people's lives during that period.

It is evident that color, being a crucial component of images or drawings, can convey significant information and aid in understanding the content of pictures. For instance, Figure 1.1a illustrates scenes from women's daily lives around 2000 years ago in China [1]. It shows that people during that time favored black and red more than

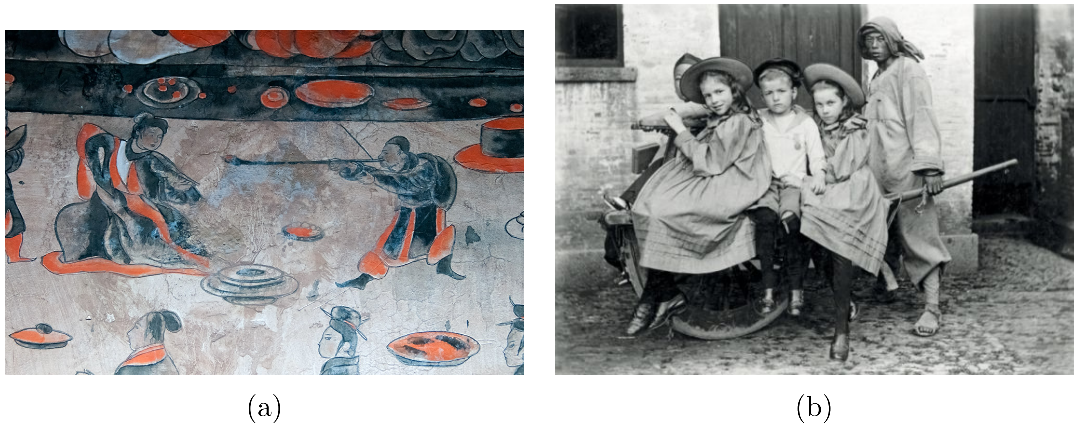


Figure 1.1: (a) A mural crafted during the Eastern Han Dynasty (25-220 AD), was found in Zhengzhou, Henan province, China [1]. (b) A photograph [2] taken by Lucy Bird Isabella in 1898.

other colors. Without this color information, it would be difficult to apply modern understanding to interpret the aesthetics of ancient people. In another example, Figure 1.1b, the image is captured without any colors, making it challenging to present the world in a more lifelike and vivid way.

In recent years, the rapid advancement of deep learning technologies has revolutionized many image processing domains. Tasks such as image denoising [3, 4], image super-resolution [5, 6], and image inpainting [7, 8] can now be automatically addressed by neural networks. Similarly, image colorization has also seen significant improvements due to the development of deep learning. While browsing the web, I came across a video titled "I used machine learning to restore the recorded Beijing a century ago" [9]. The addition of color makes the image content more vivid, lively, and captivating. The post-processed colors added to these old video footages, which have never been seen in color before, sparked my interest in related techniques.

The technology used in the video [9] is derived from Deoldify [10], an open-source, fully-automatic colorization method that uses an end-to-end CNN [11] based approach, producing impressive results on both images and videos [12]. Figure 1.2 shows an example of an image colorized using this method. There are similar methods [13–16] for adding color to grayscale images that may involve some manual intervention.

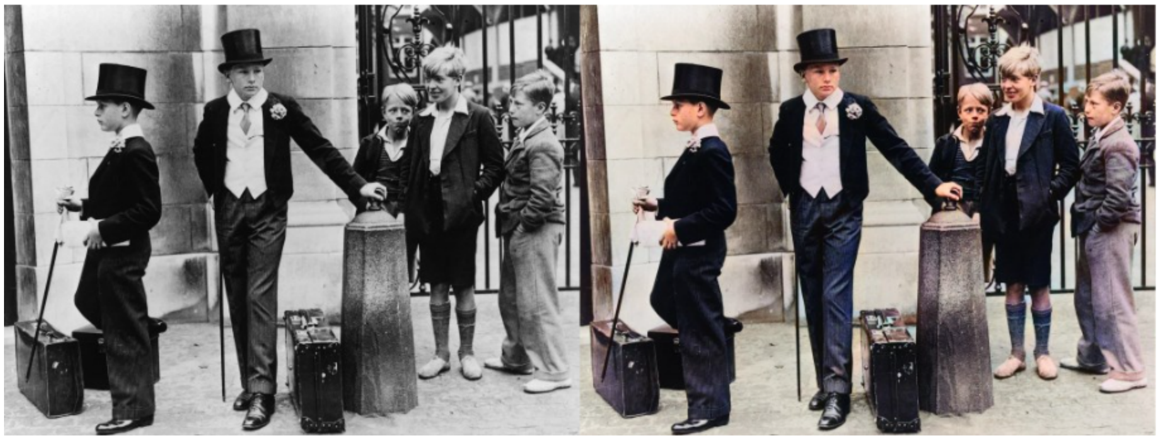


Figure 1.2: An image colorization example by Deoldify [10], where the left one is the input and the method will automatically output a colorized version on the right.

In these cases, users can specify the desired colors for specific objects or regions within an image.

Although deep neural networks have shown impressive results in grayscale image colorization, adding colors to sketches warrants further exploration. Motivated by the concept of colorizing historical images, many of which are associated with people or faces, we choose to concentrate on face sketch colorization rather than ordinary sketches for the following reasons:

(1) Each face has a unique set of characteristics, such as skin tone, eye color, and hair color, making face sketch colorization a more personalized task. The capacity to accurately colorize faces can yield more captivating and visually appealing outcomes compared to general sketches.

(2) As previously mentioned, faces consist of intricate details and unique characteristics, which render the colorization process more complex and challenging. Tackling these challenges successfully and preserving facial features accurately can be fascinating and rewarding pursuit.

(3) Hand-drawn face sketches also play a significant role in our lives. For instance, surveillance cameras cannot cover every area in our lives, and criminal suspects may still evade capture. In such cases, witness testimony is still vital to identify the

suspects. In particular, a forensic sketch artist draws a sketch of the suspect based on the eyewitness's description to assist the police to solving the crime.

(4) Lastly, the face sketch colorization problem has drawn attention from the computer vision community. We find that adding colors to a face sketch while maintaining its geometry is an interesting and meaningful research topic, and encourages innovation and breakthroughs in the field.

Thus, in this thesis, we focus on developing end-to-end face sketch colorization. Our goal is to add plausible colors and authentic textures onto a face sketch while the user can provide a real face image as the color source. As the saying states 'a picture is worth a thousand words,' we believe that images can more effectively communicate the user's desired with greater precision compared to that of words or other means. The method of using one image to guide and control the attributes of another image is referred to as "exemplar-based image translation." In Section 1.2, we explore the similarities and differences between image colorization and image translation, and their relationships with our sketch colorization task.

1.2 Image colorization and image translation

Image colorization typically refers to the process of adding colors to gray-scale images. Non-parametric methods, such as the one presented by Gupta et al. [17], assign colors to the extracted super-pixels of a given grayscale image after performing cascade feature matching with a reference image. The effectiveness of this method depends on the quality of image segmentation. As shown in Figure 1.3, when the elements and segments in the image are small or complex, the colorization results are less accurate and has some colorization artifacts, as demonstrated by the color leakage within the red circles.

Image translation involves changing a given source image into a variety of target styles. There are no restrictions on the types of these images, as the content in the source and target images can be interchanged. Traditional methods for this task

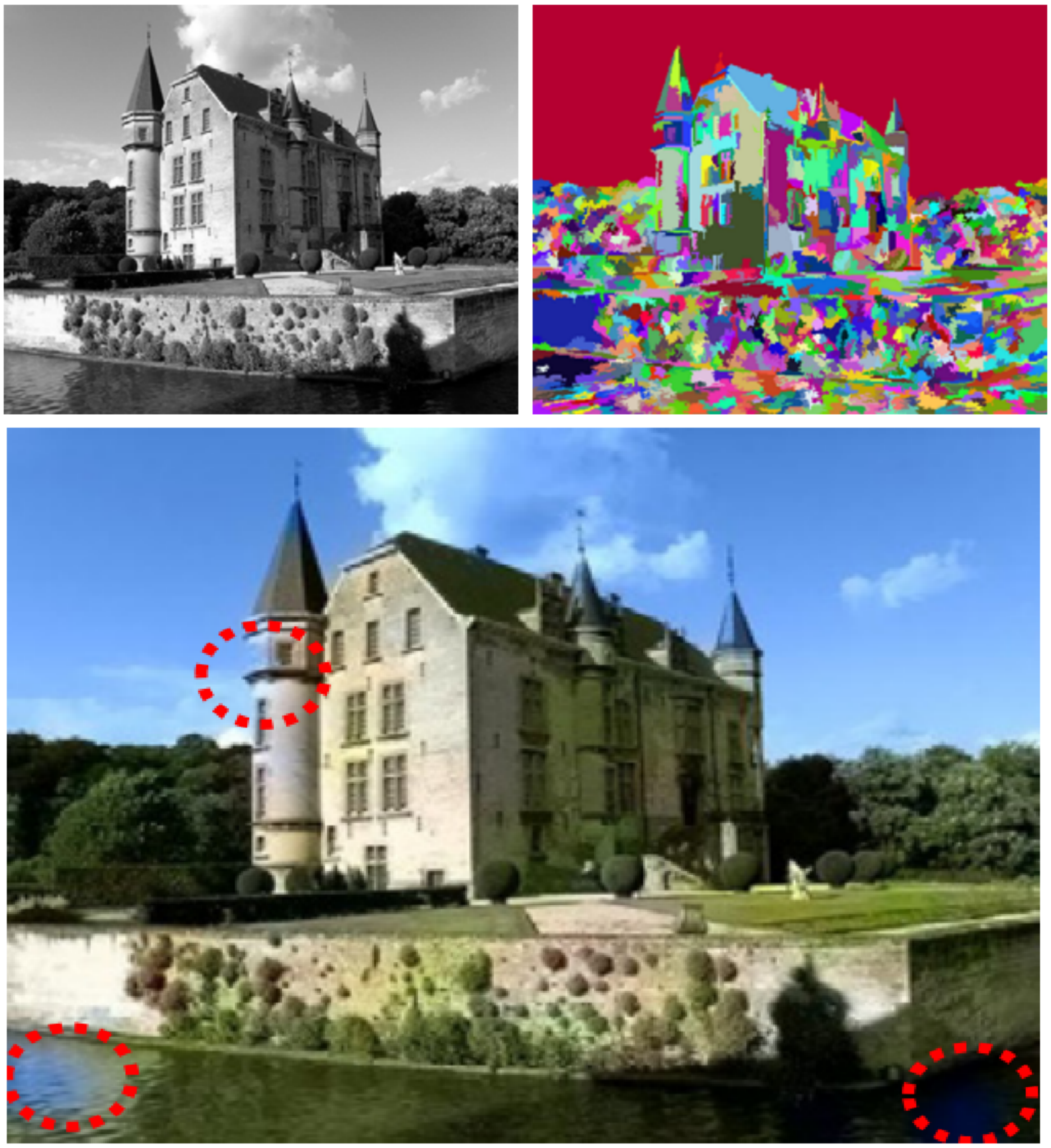


Figure 1.3: Examples of image colorization by a non-parametric method [17], first row shows the the given input gray-scale image (left) and the super-pixel segmentation (right), the second row is the colorized result. The circled region, which has a reddish tint, shows the colorization artifacts.

usually translate images between different styles by matching high-dimensional image features using algorithms such as nearest-neighbor feature search [18] or Markov Random Field [19]. For example, in Figure 1.4, the second row illustrates the conversion of a pencil-drawn face sketch into a photo of the corresponding person. However,

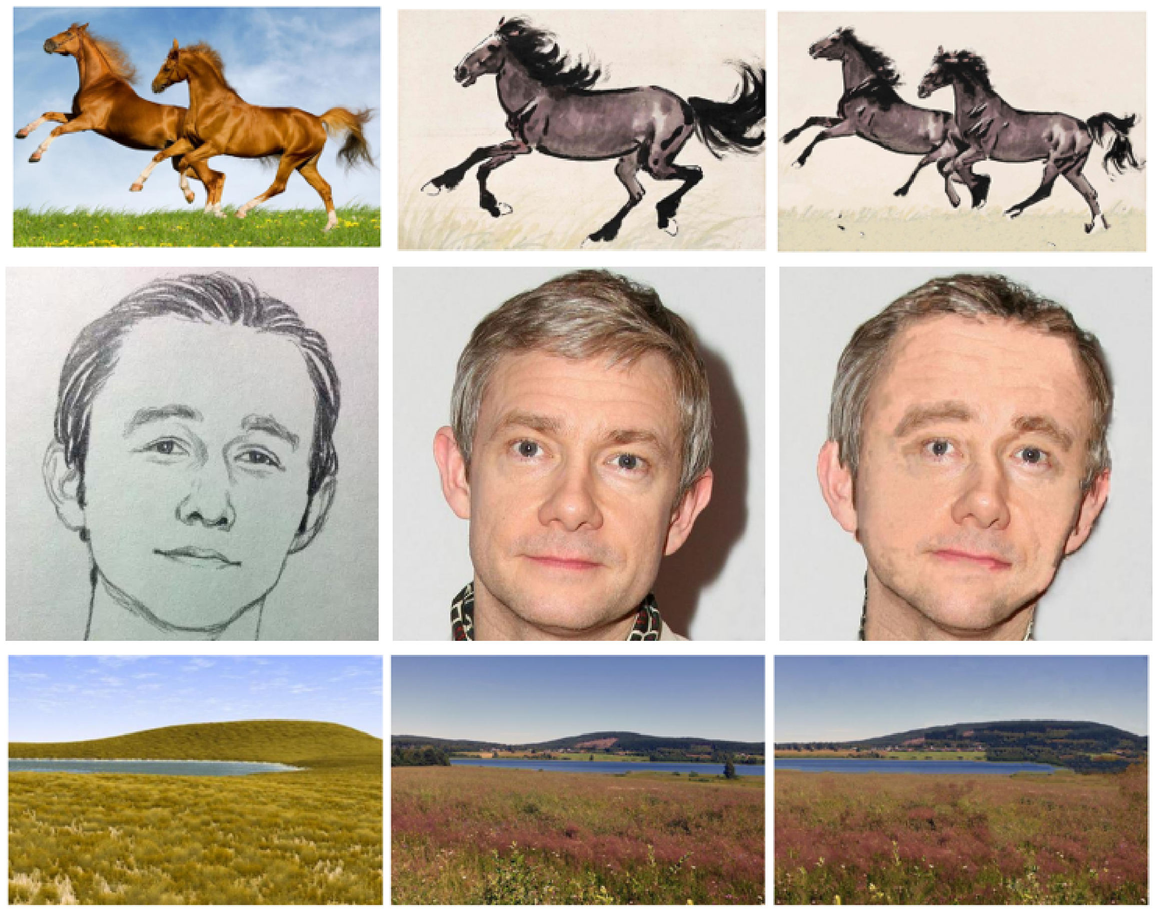


Figure 1.4: For all three rows, the left column shows the source images, the middle column the reference style images and the right column the translated images with style from the middle column and content from the left column [18].

these algorithms need a similar reference image to obtain an acceptable translation outcome.

With the advancement of Machine Learning and Convolutional Neural Networks (CNNs) [11], image translation also evolved into what is now called Neural Style Transfer [20]. As a result, the style image is no longer requires to be similar to the source image. As demonstrated in Figure 1.5, the style image can be entirely distinct from the source image.

Both tasks, colorization and style translation mentioned above have one commonality, which is translating a given image from one form to another. However, the primary difference lies in the domains of translation; the former task translates be-

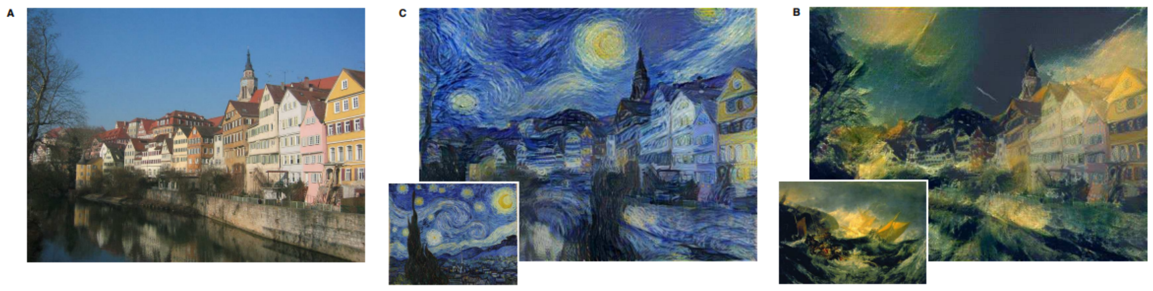


Figure 1.5: Examples provided by Neural Style Transfer [20], where the leftmost image is the original source image, while the middle and right images are synthesized versions with corresponding style images displayed in their lower-left corners.

tween grayscale and color domains, with the resulting image maintaining the same luminance information. In contrast, the latter task permits translation between any two domains, which can lead to noticeable content changes compared to that of the input images. For example, in the last row of Figure 1.4, the transferred image successfully adopts the landscape style in the middle but also introduces new objects not present in the source image, such as flowers and a distant house by the lake. Similarly, in Figure 1.5, the stylized image captures the distinctive style of the artist's painting, but the river becomes less recognizable.

Therefore, colorizing sketches can be seen as a sub-task that combines ideas from both image colorization and translation. Similar to image translation, it involves filling in the missing luminance and chrominance information while translating solely between the sketch and color domains. However, like image colorization, it also demands that the generated image has colors that strictly follow the sketch contours without distortions or changes. Consequently, our goal is to develop a method that is less restrictive, and more versatile for colorizing face sketches.

1.3 Strength of deep learning methods for image translation

In this thesis, the goal is to solve the problem of face sketch colorization using a deep learning based model. It is noteworthy that deep learning-based approaches for image translation have gained significant attention in recent years due to their advantages, which include:

- Deep learning models can generate realistic and visually appealing face images using large amounts of data to learn the complex relationships between input sketches and the corresponding colored images. Figure 1.6 shows some examples.

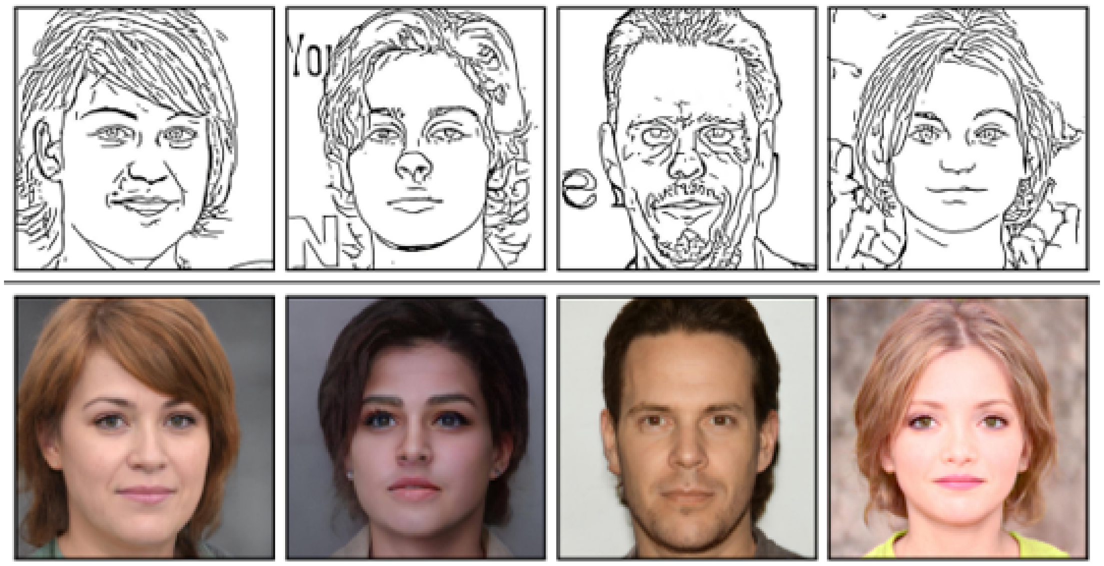


Figure 1.6: Multi-modal synthesis of a single input performed by Richardson, et al. [21] with a resolution 1024 x 1024. The top row shows the sketch input and the bottom row shows the synthesized image.

- Recent deep learning models have unique mechanisms that enable them to automatically learn and extract relevant features from the input data, and to fuse these features together without requiring manual feature engineering. This approach allows for a more efficient and effective way to translate an image into the desired domain, as the model can automatically identify and use the most relevant information. More details are given in the Related Works Section.

- Some deep learning-based models, like normalizing flow [22, 23], demonstrate lossless preservation properties that guarantee the preservation of essential information from the original image during the encoding and decoding processes. For example, in the research by An et al. [24], the authors use a normalizing flow for style transfer, while maintaining the capability to recover the original content image losslessly from the stylized image. The geometry information of a content image can also be preserved.

However, these approaches are not suited for face sketch colorization. For CNN-based generative models like Generative Adversarial Networks (GAN) [25], these methods often fail to preserve the geometry of the sketch. For instance, as shown in Figure 1.6, the proposed GAN-based method [21] samples colors from a random distribution while using the sketches in the first row as input. While all synthesized images resemble human faces, they do not accurately respect the input sketches. Other methods, such as normalizing flow-based methods, generate unsatisfactory results for the sketch-to-image colorization task, because they generally struggle to establish an effective connection between the two image domains, which have a domain gap. We discuss these deep learning models in more detail in Chapter 2.

In this thesis, we address the face sketch colorization problem by incorporating ideas from both CNN-based and normalizing flow-based methods. We employ a normalizing flow-based model in our model architecture, specifically designed to losslessly preserve essential information during the encoding and decoding processes. To supply color information, we adopt an exemplar-based strategy, enabling users to control colors by selecting their preferred color images to guide the final colorization results. Our model adapts to unpaired images, allowing the sketch image to accept and integrate color information from entirely distinct sources.

To effectively add color to a face sketch, we follow many CNN-based methods [21, 26–30] and introduce a new Feature Aggregation Module (FAM) to adaptively merge the intermediate latent features extracted by our model. This module enhances the colorization accuracy and quality. Our flow-based model workflow begins by taking a face sketch and a real color face image as input. These images are subsequently transformed into high-dimensional latent variables and gradually fused using our FAM. The model then generates the final colorized face sketches based on the combined information. In practice, we design our model's architecture to use less computational resources, resulting in reduced GPU memory usage and faster training compared to that of other flow-based models.

In our experiments, we use a comprehensive method to qualitative and quantitative evaluate and compare our method with methods based on GANs and on normalizing flows. Since we are the first to use a normalizing flow-based model for face sketch colorization, we tried our best to achieve more satisfactory and realistic results. The results show that our proposed method can preserve the sketch geometry and faithfully adapt the colors of the reference image.

1.4 Thesis objectives

The objective of this thesis is to develop a normalizing flow-based method that accurately colorizes a face sketch while remains faithful to the input sketch geometry. Our proposed model can better capture the content of face sketches and accurately colorize them using various input colored face images. In summary, our main contributions include the following three main aspects:

- We propose a memory-efficient normalizing flow-based model that accurately follows the geometry of a given face sketch to create a properly colorized face image while using less computational resources compared to traditional flow-based architectures.

- We introduce a new FAM in our model to adaptively merge unpaired color and sketch features. An ablation study further demonstrates the effectiveness of this new module.

- We compare our flow-based model with existing CNN-based and flow-based image translation methods using comprehensive qualitative and quantitative experiments. The results demonstrate that our approach is more effective to address the face sketch colorization problem.

1.5 Thesis organization

In this chapter, we present our motivation in address the problem of face sketch colorization and discuss the main challenges of existing methods. As an overview of this thesis, our approach combines ideas from various deep learning-based models to address this problem, leading to better performance than previous works. The rest of the thesis is organized as follows:

In Chapter 2, we introduce the background knowledge in sketch colorization as well as the deep learning based methods for sketch colorization related problems.

In Chapter 3, we provide a detailed explanation of our model's design and discuss the procedure of data pre-processing.

In Chapter 4, we carry out thorough qualitative and quantitative experiments, including a user study, to demonstrate the effectiveness of our method.

In Chapter 5, we conclude the entire thesis, and show the limitation of our flow-based model that can be addressed in the future.

Chapter 2

Background

As we introduced before, deep learning plays an important role in image colorization and image translation, with various deep learning-based methods demonstrating unique strengths in solving these tasks. Our research focuses on leveraging the advantages of these deep learning methods for face sketch colorization, enabling an accurate mapping of geometric information to the colorized image while adopting appropriate colors from the reference image.

In this chapter, we introduce the background information relevant to our work. In Section 2.1, we introduce several deep learning-based image colorization methods. Then, in Section 2.2, we investigate the mechanisms and strategies employed in GAN-based models for controlling the results in unpaired image-to-image translation. Lastly, in Section 2.3, we discuss normalizing flow and its applications in image processing.

2.1 Deep learning based image colorization

Due to the limited generalizability of non-parametric methods, deep learning-based models for end-to-end colorization have become the favored approach. These methods use the CIELAB color space (Lab), which divides an RGB image into a perceptual lightness (the L channel) component and two distinct colors components (the ab channels). In this setting, the L channel serves as the input to the model, while the ab channels represent the target information that the model needs to predict.

Initially, some researchers [31–33] find that using simple plain networks can generate diverse colors given a gray-scale image; however, the final colors are not controllable. In response, subsequent research efforts focus on developing different methods to control the generated colors.

Some methods require users to provide additional information, such as color dots or strokes [15, 16], at specific locations in the image. For example, as displayed in Figure 2.1, users can add green or red colors to the pepper to achieve the desired colorized result. Nonetheless, these methods often rely on user input to achieve better quality in the output.

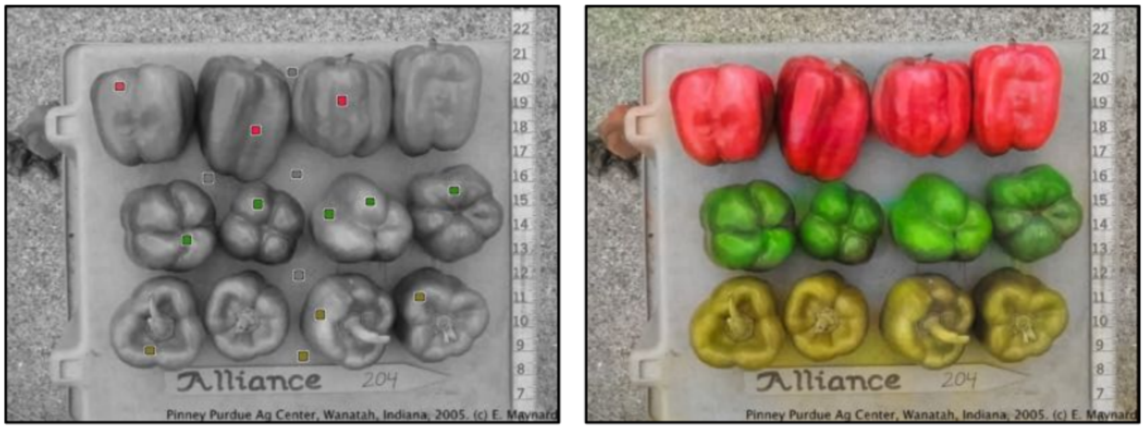

Alliance
204
Pinney Purdue Ag Center, Wanatah, Indiana, 2005. (c) E. Maynard
Alliance
204
Pinney Purdue Ag Center, Wanatah, Indiana, 2005. (c) E. Maynard

Figure 2.1: Examples provided by Zhang, et al. [15]. The left image represents the synthesized input, while the right image shows the colorized result.

Other methods enable users to control colors using text descriptions. For example, methods proposed by Manjunatha, et al. and Bahng, et al. [13, 14] can generate color images using descriptive words, as shown in Figure 2.2. As well, some of the most latest and popular models, including Stability AI [34], DALL·E 2 [35], and Midjourney [36], merge text and images into a shared high-dimensional space, using diffusion model to synthesize the results. However, using textual descriptions to control the image synthesis process has been shown to be imprecise and prone to errors. In more complex scenes with numerous small objects or unique items, object

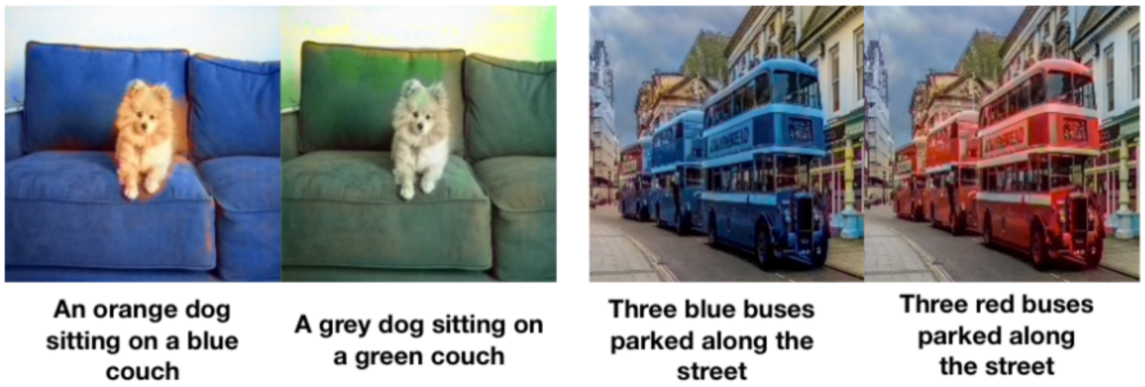

An orange dog sitting on a blue couch
A grey dog sitting on a green couch
Three blue buses parked along the street
Three red buses parked along the street

Figure 2.2: Examples provided by Manjunatha, et al. [13], where they can colorize an image based on the text provided below.

detection becomes more challenging, and detailed textual descriptions may introduce ambiguity, potentially failing to accurately represent the desired results of the user.

To simplify the colorization task and make it more straightforward, recent attention has shifted towards using exemplar-based techniques to control how the color styles are applied to grayscale images. He, et al. [37] propose a deep learning-based method that addresses this problem by using a deep neural network, VGG19 [38], to align the exemplar image and the source image at the deep feature level, and then reconstruct a coarse aligned reference image for further colorization. A similar strategy can be found in the work by Xu, et al. [39], who use an Encoder-Decoder as a style transfer network. Their approach produces a rough reference color map in the ab channels, which along with the source image, is input to another Encoder-Decoder [40] to obtain a refined colorization result. It is important to note that these methods all use the Encoder-Decoder framework for image colorization. As shown in Figure 2.3, the encoder extracts low- to high-level features, while the decoder reconstructs the colorized image using these features. This hierarchical representation helps capture both local and global structures in the image, resulting in more accurate and visually appealing colorization outcomes.

We observe that the previously mentioned methods have limitations in adding col-

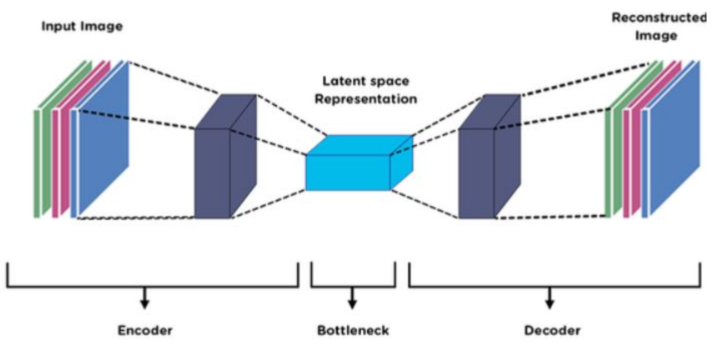


Figure 2.3: An example of Encoder-Decoder architecture

ors to the luminance information. Typically, these approaches output only the 'ab' channels of the 'Lab' space, and the final colorized images are created by combining the input grayscale image with the output color maps. However, if the input is a coarse face sketch, traditional colorization methods struggle to add suitable colors. Directly combining the sketch image with color maps may result in perceptual disharmony for viewers. Instead, our approach aims to enable the model to directly output colorized face sketches. Similar to the earlier methods, our model also adopts the Encoder-Decoder framework and the exemplar-based training strategy. In the upcoming sections, we focus on designing a control mechanism within our model that can adaptively apply color information from different images to colorize face sketches, and on selecting an appropriate network for our Encoder-Decoder architecture.

2.2 Unpaired image-to-image translation

Recently, Generative Adversarial Networks (GANs) have seen great success as generative models because of their realism and state-of-the-art performance in image translation tasks. In this section, we first introduce the architecture of GAN, and then explore exemplar-based image translation methods that use GANs and their variations. We also discuss how these approaches intelligently manage style and content information to create visually pleasing stylized images.

2.2.1 Generative Adversarial Networks

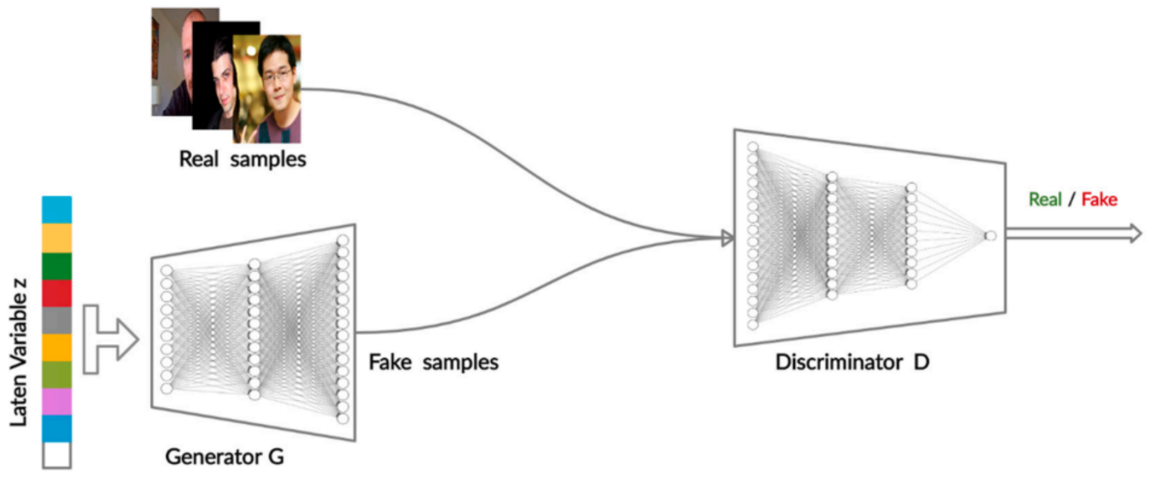


Figure 2.4: Example that shows the architecture of GAN [41].

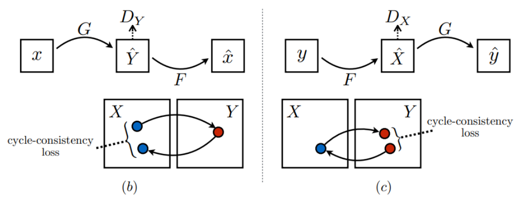
The Generative Adversarial Network was first proposed by Goodfellow, et al. [25] in 2014. It has two main parts: a Generator (G), also known as the Decoder, and Discriminator (D). In the context of image translation, the generator's goal is to create fake images from a specific latent space that look real. On the other hand, the discriminator's job is to clarify fake images made by the generator from the real images from the dataset.

Their roles can be seen as competing with each other. During training, the generator is optimized by minimizing the error between the fake image and the real image, while the discriminator is optimized by maximizing the chance of correctly telling apart the labels on both real images and fake images made by the generator. The latent variables can come from either a random distribution or a specific distribution given by an encoder [41]. For many image translation methods, a clear sign that the models are GAN variants is that they all include a discriminator without exception [41].

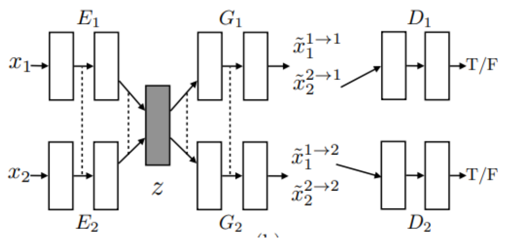
2.2.2 Cycle consistency in control contents and styles

Unsupervised training is an early approach for dealing with unpaired image-to-image translation problems, allowing for high-quality translations even when there is limited training data. Initially, Jun-Yan, et al. [42] introduce the CycleGAN, which uses two identical generators to translate images between two different domains and two identical discriminators as competitors. As shown in Figure 2.5a, generator G maps images from domain X to Y, while generator F maps images from domain Y to X. Cycle consistency training ensures that translation is reversible by successively mapping images back to their original domain using the reconstruction error. The implementation of generator G follows the Encoder-Decoder architecture in Figure 2.3, encoding each domain into separate latent spaces.

UNIT [43], shown in Figure 2.5b, which is one of the variants, encodes images into a shared latent space and encourages the model to learn the joint distribution of multi-domain images in an unsupervised manner [41]. DIRT [44] also encodes images into a shared space, but separates it into content space and domain-specific attribute spaces for a more stable and accurate representation of synthesized images. However, the content in synthesized results can be easily changed compared to the source input. As shown in Figure 2.6, the face image successfully captures the style, but the content changes to another similar person. Similarly, the trees in the first row transform into a mountain due to their geometric resemblance.



(a)



(b)

Figure 2.5: (a) CycleGan architecture [42]. (b) UNIT architecture [43].

For our task, the cycle consistency training strategy effectively ensures that syn-

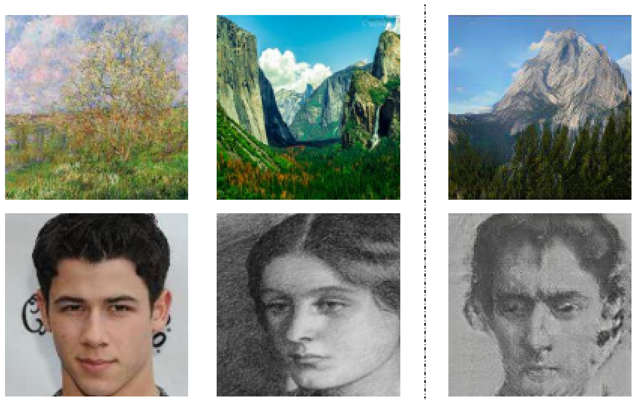


Figure 2.6: Image translation examples from UNIT [44]. The left column represents the content domain, the middle column displays the attribute or style domains the method aims to capture, and the right column presents the final synthesized results.

thesized results remain within the image domain. However, to produce images that closely adhere to content geometry, simply using this mechanism is not enough. We need to investigate additional techniques that can control geometry more accurately. Additionally, the cycle training strategy involves more encoding and decoding processes compared to supervised learning, leading to higher computational costs during model training.

2.2.3 Feature Aggregation Module for contents and styles control

Normalization based modules

In order to achieve more precise control over both style/colors and content/geometry, some methods use normalization and its variants to fuse different deep latent features. For example, Huang, et al.[28] first propose Adaptive Instance Normalization, building upon Instance Normalization [45]. This technique adjusts the mean and variance of the input content to match that of the input style, maximizing consistency with the original geometry [28]. The calculation step can be expressed by

$$AdaIN(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y), \tag{2.1}$$

where x denotes the input content, $\mu(.)$ the mean of (.), $\sigma(.)$ the standard deviation of (.), and y the input style. In Eq 3.1, it can be seen that, the first step is to shift the mean to zero and normalize the standard deviation of x to one, and the second step is to scale the standard deviation and to shift the mean to that of y.

Park, et al. [27] extend the above ideas by developing a module called Spatially-Adaptive Denormalization (SPADE), depicted in Figure 2.7. This module injects content information to latent variables derived from random distributions or style images. Additionally, normalization-based feature aggregation modules can also combine image and non-image features effectively. For example, the method by Yang et al.[26] encodes various refinement levels as numbers into means and standard deviations represented in a high-dimensional level to control how much sketch geometry information should be preserved in the synthesized colorful image.

To create a more generalized calculation method for aligning style and content features, various studies [46–48] encode style or content information into the weights and biases used in convolution operations on instance denormalized latent variables. In the context of sketch image translation, several recent works [21, 49–51] have successfully captured color or style information and generated visually appealing, realistic face images by leveraging this mechanism.

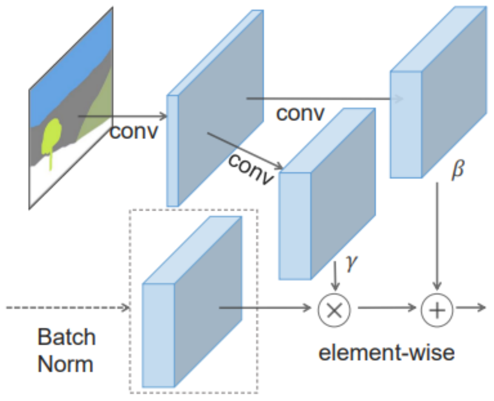


Figure 2.7: An overview of SPADE [27] reveals that it shares similarities with AdaIN [28] in normalizing a latent variable using two vectors, γ and β. However, unlike the mean and standard deviation, γ and β are learnable modulation parameters that are spatially aligned with the target latent variable.

Attention based modules

With the popularity of attention in NLP tasks [52], many researchers realize that it can also be applied in image processing task. The basic architecture of the attention module is illustrated in Eq2.2. Here, Q, K, and V represent high-dimensional image features. Typically, Q and K come from different image domains, resulting in an attention map $softmax(\frac{QK^T}{\sqrt{d_k}})$, which serves as a weight map indicating the correlation between the two distinct features. The term $\sqrt{d_k}$ scales the dot products to counterbalance the small gradient produced by the softmax function. Finally, the feature map V is multiplied by the weight map, determining the amount of information carried by feature V for subsequent calculations.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.2}$$

In sketch to image translation, Lee et al. [30] successfully apply an attention module, which they named 'Spatially Corresponding Feature Transfer (SCFT)', as depicted in Figure 2.8. In this module, V_r and V_s, respectively, represent groups of latent feature maps encoded from a color reference image and a sketch image. The fused latent variable, V_r^*, combines the sketch and color features, and is element-wise summed with V_s to decode into the final colorized image. Similar applications can be found in the works of Zhang et al. and Wu et al. [53, 54], in which they align features through an attention map and use these features to guide image translation. Additionally, Liu et al. [29] develop a method, 'AdaAttN', that combines both normalization and attention. As shown in Figure **??**, the approach first computes an attention map, then uses it to calculate the mean and standard deviation with style feature F_s^x, and then performs Adaptive Normalization with content feature F_c^x.

We notice that while the feature aggregation modules discussed earlier can successfully generate stylized images for GAN-based methods, these methods often struggle to preserve the geometry of the given sketch image, which is due to the variations during image generation. For instance, in Figure 2.10, Liu et al. [51] use AdaIN to

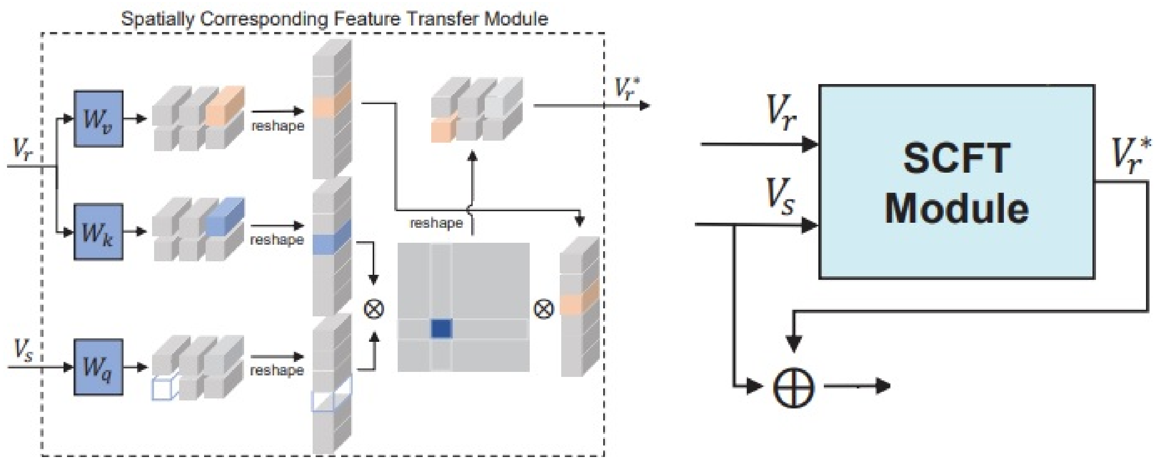


Figure 2.8: Architecture of Spatially Corresponding Feature Transfer (SCFT) [30].

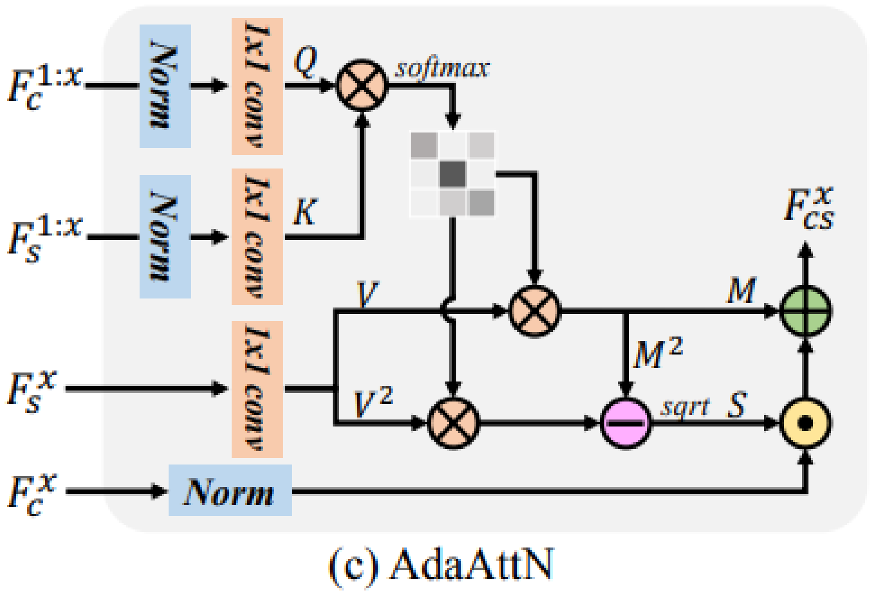


Figure 2.9: Architecture of 'AdaAttN' [29].

fuse style information with the coarse face sketch. As the reference image in the first row changes, the beard of the man in the second row also changes. While the origin of this issue is unclear, it has not gain much attention from researchers popular, and it is still a relatively quiet area of investigation.

Therefore, our goal is to develop a model for our Encoder-Decoder architecture that can precisely infer information from latent variables while maintaining the face sketch geometry as much as possible. Furthermore, to effectively fuse sketch with color information, we design our feature aggregation module based on attention mechanism [52]. We then integrate this module into our Encoder-Decoder architecture to achieve sketch colorization that remains faithful to the sketch and accurately captures the reference colors.

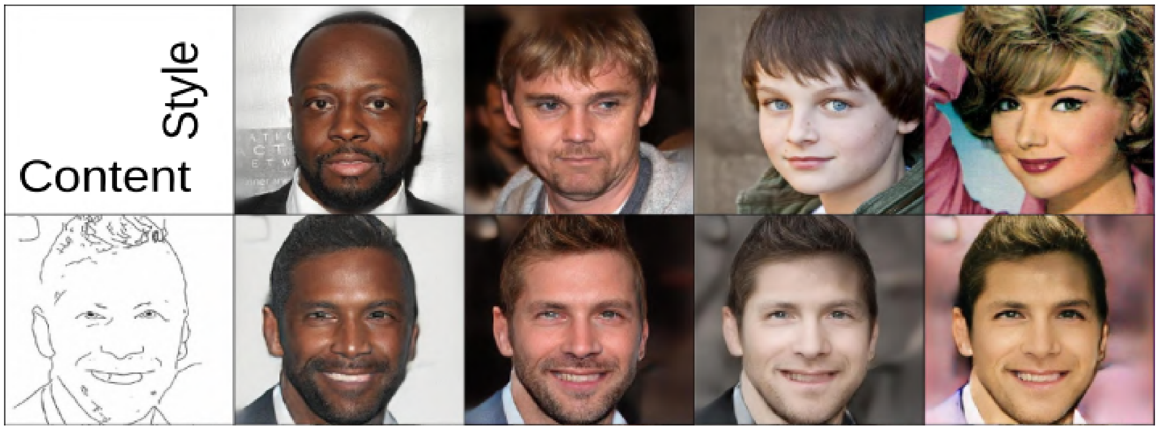


Figure 2.10: Examples of sketch to face translation performed by Liu et al. [51].

2.3 Normalizing Flow

Normalizing flow [22] refers to a class of generative models that use a sequence of invertible affine transformations to map a simple base distribution, such as a Gaussian distribution, onto a more complex target distribution. By stacking more affine transformations, the model can capture more intricate details of the data distribution, allowing it to generate samples that are closer to the true data distribution. The generative process for a majority of normalizing flow-based models [22–24, 55–59] can be expressed as follows:

$$x = G_\theta(z) \tag{2.3}$$

where z is the latent variable that follows a distribution. For example, a Gaussian distribution with $z \sim N(0, 1)$. The model G_θ is invertible, which means that a given datapoint x can be encoded into a random variable z using the following equation:

$z = G_\theta^{-1}(x)$. As G_θ is composed of a series of affine transformation, we can also denote G_θ as a composition of K transformations:

$$G_\theta = g_\theta^1 \circ g_\theta^2 \circ \cdots \circ g_\theta^K \tag{2.4}$$

where g_θ^i is a single transformation step, for $i \in \{1 \cdots K\}$. The relationship between x and z can also be denoted as:

$$x \overset{g_\theta^1}{\longleftrightarrow} h_1 \overset{g_\theta^2}{\longleftrightarrow} h_2 \cdots \overset{g_\theta^K}{\longleftrightarrow} z \tag{2.5}$$

where h_i is any intermediate latent variables for $i \in \{1 \cdots K\}$.

The advantages of flow-based methods are summarized below: (1) They offer more stable training compared to typical generative models such as GANs. (2) They provide exact data encoding and latent variable decoding process, allowing datapoints to be fully represented in a latent space. (3) They enable exact log-likelihood evaluation, which can estimate expressive distributions of any given data space.

In recent years, Glow (Generative Flow) [23], a normalizing flow-based generative model, has gained popularity, and many subsequent models [55–58] have demonstrated impressive performance in various image generation-related tasks, such as image super-resoltion [55, 60], image denoising [58] and image hiding [56]. As for the architecture of normalizing flow, we select Glow [23] as the most representative model to introduce. Table 2.1 illustrates the three fundamental computational steps in both forward and reverse orders, which are the main components of Glow [23]:

- The first one is the Actnorm layer, which serves a similar purpose as batch normalization [61] in addressing the vanishing gradient problem when training deep models. This layer uses a scale (`s`) and bias (`b`) parameter to perform an affine transformation on a given latent variable $x_{i,j}$. These parameters are initialized as zeros and function as general trainable parameters, independent of the data [23].

Description	Function	Reverse Function
Actnorm.	$\forall i,j : \mathbf{y}_{i,j} = \mathbf{s} \odot \mathbf{x}_{i,j} + \mathbf{b}$	$\forall i,j : \mathbf{x}_{i,j} = (\mathbf{y}_{i,j} - \mathbf{b})/\mathbf{s}$
Invertible 1x1 convolution W:[c x c].	$\forall i,j : \mathbf{y}_{i,j} = \mathbf{W}\mathbf{x}_{i,j}$	$\forall i,j : \mathbf{x}_{i,j} = \mathbf{W}^{-1}\mathbf{y}_{i,j}$
Affine coupling layer [22]	$\mathbf{x}_a, \mathbf{x}_b = \mathit{split}(\mathbf{x})$ $(\mathit{log}\ \mathbf{s}, \mathbf{t}) = NN(\mathbf{x}_b)$ $\mathbf{s} = \mathit{exp}(\mathit{log}\ \mathbf{s})$ $\mathbf{y}_a = \mathbf{s} \odot \mathbf{x}_a - \mathbf{t}$ $\mathbf{y}_b = \mathbf{x}_b$ $\mathbf{y} = \mathit{concat}(\mathbf{y}_a\ , \mathbf{y}_b)$	$\mathbf{y}_a, \mathbf{y}_b = \mathit{split}(\mathbf{y})$ $(\mathit{log}\ \mathbf{s}, \mathbf{t}) = NN(\mathbf{y}_b)$ $\mathbf{s} = \exp(\log\ \mathbf{s})$ $\mathbf{x}_a = (\mathbf{y}_a - \mathbf{t})/\mathbf{s}$ $\mathbf{x}_b = \mathbf{y}_b$ $\mathbf{x} = \mathit{concat}(\mathbf{x}_a\ , \mathbf{x}_b)$

Table 2.1: Three components of Glow: ActNorm, an invertible 1x1 convolution layer and an affine coupling layer. For given variables `x` and `y` in the shape of $[h \times w \times c]$ with spatial dimensions (h, w) and channel dimension c, (i, j) denotes spatial indices [23]. For the symbols, $\odot$ denotes the matrix multiplication, *split* the equally channel level split operation on given latent features x or y, NN the any neural networks, *exp* the exponential operation and *concat* the concatenation operation on two latent features.

- The second component is an invertible 1×1 convolution, which uses a learnable random rotation weight matrix `W` as parameter that performs permutation on a given varibale $x_{i,j}$. The input and output channels of this variable remain the same. The purpose of this component is to ensure that all channel information can adaptively participate in training, resulting in better generalization quality.

- The final component is an affine coupling layer. It first splits the channels of the received variable `x` into two equivalent variables x_a and x_b, each with a shape of $[h \times w \times \frac{c}{2}]$. One of these variables, such as x_b, is passed to a simple neural network `NN` to estimate the element-wise scaling parameter s and element-wise translation parameter t for the affine transformation. The other part of the variable, x_a in this case, undergoes an affine transformation using these element-wise parameters. Finally, the transformed x_a is concatenated with x_b to form the output of the affine coupling layer.

To simplify the description, researchers typically combine the aforementioned three modules into a single module consisting of an actNorm, an invertible 1x1 convolution, and an affine coupling layer in sequence. We refer to this module as the 'Coupling Block', while in Glow, it is called the 'flow'. Furthermore, due to limited computational resources, Glow [23] and many subsequent works [24, 55–59, 62] incorporate these 'Coupling Blocks' into a multi-scale architecture. This requires the normalizing flow-based model to include both downsampling and upsampling layers. In this context, the downsampling layer must also be invertible, so that the reverse process of the downsampling operation is upsampling. Glow [23] addresses this problem by using a Squeeze layer, which halves the spatial size of the latent variable h by reshaping each spatial 2 times 2 neighborhood into the channel dimension [55]. Ardizzone, et al. and Jing, et al.[57, 63] use wavelet transformation to perform downsampling [64] which is also an invertible operation. Christian, et al. [65] propose a learnable orthogonal kernel and use it in a convolution layer for performing invertible downsampling.

In works focusing on image colorization or translation, Ardizzone et al.[63] use a conditional invertible neural network for grayscale image colorization, avoiding issues such as mode collapse and unstable training often encountered in GANs. However, their approach is limited to adding chrominance information to luminance images. An, et al [24] incorporate an unbiased feature transfer module into a normalizing flow-based architecture, enabling style transfer in a lossless and unbiased manner. While their method effectively preserves the geometry of the content image, it has difficulty translating a face sketch to a realistic image. Sorkhei et al. [59] also use a normalizing flow-based architecture and carry out high-resolution semantic segmentation map-to-real image translations, but their task is restricted to paired data translation.

It is evident that existing flow-based methods are not suitable for our face sketch colorization task, which requires unpaired color face images to guide the colorization process. Additionally, the significant domain gap between face sketches and real face images poses a challenge for these methods to effectively establish a correlation between the two types of images. Nonetheless, the precise encoding and decoding mechanisms of normalizing flow are crucial for preserving image information. For instance, in Figure 2.11, ArtFlow [24] demonstrates near-perfect preservation of the original image's content information when transferring different styles to face images. We believe that integrating a suitable feature aggregation module into normalizing flow could aid in the face sketch colorization task, resulting in a colorized face sketch with accurate, visually appealing colors that remain faithful to the sketch.

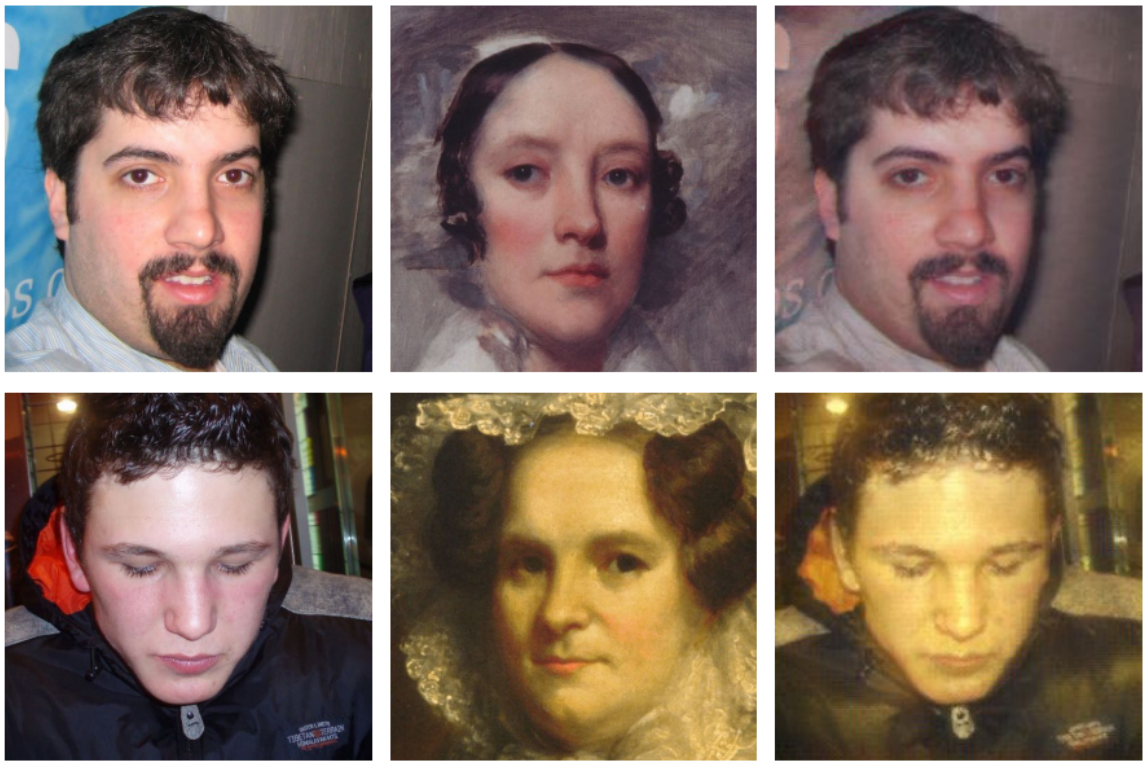


Figure 2.11: Examples from ArtFlow [24]. The left column is the content image, the middle column is the style image, and the right column is the stylized image.

2.4 Summary

In this chapter, we discuss various deep learning-based image colorization methods and analyze why they cannot be directly applied to our face sketch colorization task. We provide background information on Generative Adversarial Networks (GANs) and the modules and strategies used in GAN-based methods for controlling content or style information in a translated image. These include the cycle consistency training strategy, normalization, and attention-based feature aggregation modules. Finally, we introduce the background of normalizing flow and its related methods in image translation.

In the next chapter, we present our model architecture for face sketch colorization, which uses normalizing flow in an Encoder-Decoder manner and integrates our proposed feature aggregation module for improved adaptation between sketch and color information.

Chapter 3

Face sketch colorization method

In Section 3.1, we first introduce our model architecture, including the overall workflow and details of each module separately. Then we discuss the novelty of our model, which features a memory-efficient design in Section 3.2. Later, we introduce the data preparation process which includes data augmentation strategies and the steps on how we generate face sketches from real images in Section 3.3. In Section 3.4, the loss function is discussed followed by the implementation of the proposed method in Section 3.5.

3.1 Model architecture

The purpose of the network is to colorize a face sketch I_s given any color face image I_{ref} as the reference color image. We consider this task as a one-to-many mapping task, such that we can map these two images into their latent domains, and then restore that sketch latent feature to the domain of the color image by leveraging the advantages of normalizing flow.

3.1.1 Coupling block

Our network consists of a series of coupling blocks, along with several down samples or upsampling layers. Our coupling block is designed to perform a reversible affine transformation, with the main component being the coupling layer, first introduced by

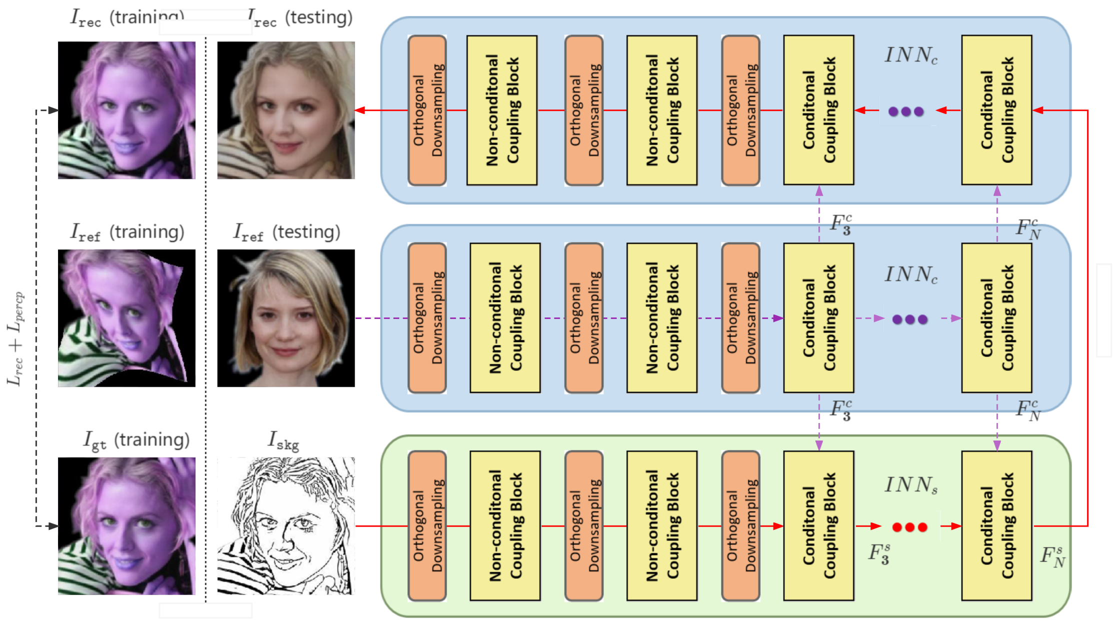
Dinh et al.[22]. We discuss more details in the following. Additionally, there are two types of coupling blocks: Conditional and Non-Conditional. The term 'Conditional' refers to a module that incorporates additional information to guide the generation of output, while 'Non-Conditional' indicates that the module generates the output based solely on the input data it receives, without relying on any external guidance or constraints.

3.1.2 Workflow overview

The workflow of our method is shown in Figure 3.1. Unlike ArtFlow, [24] which uses a shared encoder for both content and style images, we use two normalizing flows such that their forward process plays the same role as the traditional CNN-based encoder, which encodes the sketch image I_s and the color image I_{ref} into two groups of intermediate latent features X and Y, respectively. The reverse process plays the same role as a decoder which maps the latent features back to the corresponding image domains. To keep it simple, our normalizing flows, which function more like an invertible neural network, are denoted as INN_s and INN_c for taking sketch and color face images as input, respectively.

Denote $X = \{F_i^s\}_{i=1}^N$ and $Y = \{F_i^c\}_{i=1}^N$, where F_i is the output latent feature from the i^{th} coupling block, and the superscripts s and c denote sketch and color, respectively. Then for the final encoded sketch feature, we have : $F_N^s = INN_s(I_s\ ,\ \{F_i^c\}_{i=j+1}^N)$ where N is the total number of coupling blocks of each INN, and j stands for the number of non-conditional coupling blocks with latent features not used for conditioning. The set of latent features from the non-conditional coupling blocks of INN_s, is denoted as $\{F_i^s\}_{i=1}^j$.

For the decoding process, we process the last latent feature F_N^s in the reverse order of the color encoder INN_c. The goal is to accurately restore this latent feature back to the color image domain by using the weights from the color encoder. We denote this process as : $I_{rec} = INN_c^{-1}(F_N^s\ ,\ \{F_i^c\}_{i=j+1}^N)$, where I_{rec} is the output colorized



I_{rec} (training)
I_{rec} (testing)
I_{ref} (training)
I_{ref} (testing)
I_{gt} (training)
I_{skg}
$L_{rec} + L_{percp}$
Orthogonal Downsampling
Non-conditonal Coupling Block
Conditonal Coupling Block
INN_c
INN_s
F_3^c
F_N^c
F_3^s
F_N^s

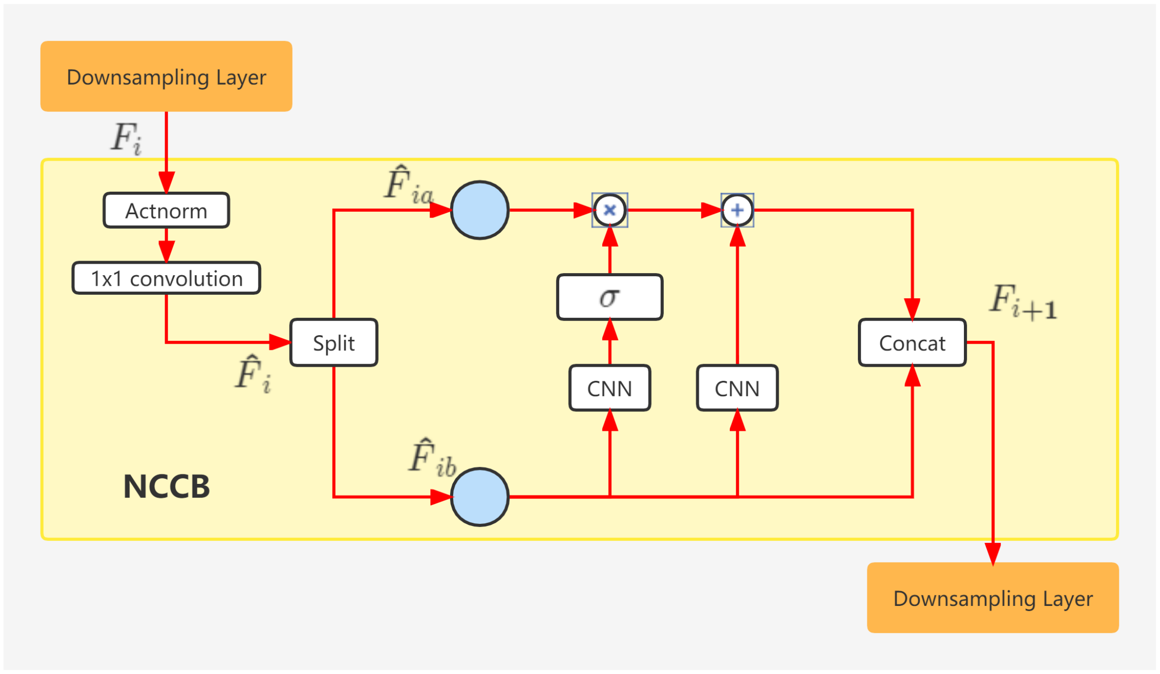
Figure 3.1: Overall architecture of our proposed network. The network in the green rectangle is our sketch encoder INN_s. The network in the blue rectangle is our color encoder INN_c. The purple dotted line indicates the color encoding process and the red line indicates the sketch encoding and decoding processes. The model in the blue rectangle is basically the same as that in the green rectangle. We separate them for clarity.

image. It is important to note that we don't need an extra branch to recreate the sketch in order ensure the colorized image adheres to the sketch's structure. This is because our normalizing flow can losslessly convert an image into a latent space and decode the latent features back to the original image domain. In this case, the sketch's geometry is preserved throughout the entire encoding and decoding processes.

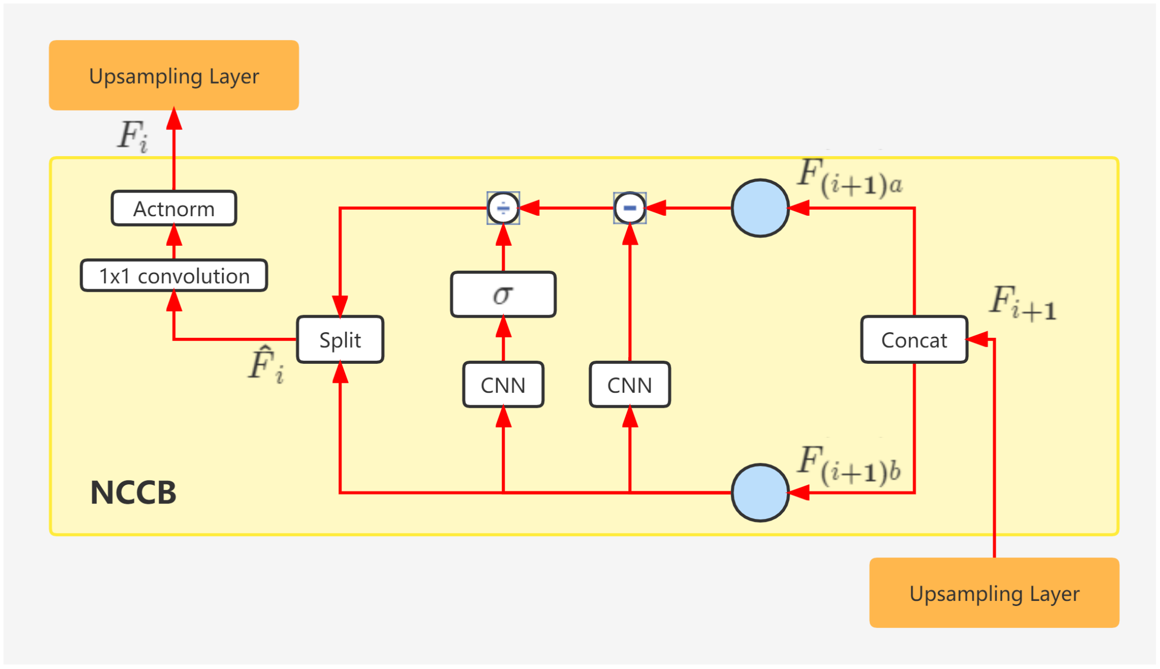
3.1.3 Normalizing flow design

Our normalizing flow has three key components: (1) a non-conditional block, (2) a conditional coupling block (see Figure 3.2), and (3) an invertible downsampling or upsampling layers.

(1) For the non-conditional coupling block ($NCCB$), we adopt the coupling block used in Glow [23] and ArtFlow [24], which consists of three layers: an *ActNorm layer (***act***)* which acts as batch normalization, a *Channel Permutation Layer* (**p**), which



(a)



(b)

Figure 3.2: The architecture of the non-condtional coupling block ($NCCB$). (a) The encoding process of NCCB where it takes a single input F_i and outputs feature F_{i+1}. (b) The decoding process of NCCB where it takes a single input F_{i+1} and outputs feature F_i.

is implemented using an invertible 1 x 1 convolution[23] to encourage every channel information to participate in the learning of deep features, and (3) a *coupling layer* (**coup**) which uses half of the latent feature information in the affine transformation of the other half. Figure 3.2 shows the detailed design of the NCCB. It is noteworthy that, regarding the architecture of this module, the only difference from Glow [23] and ArtFlow[24] is the choice of the CNN, where we use a residual block for a better nonlinear mapping [56].

Let F_i denote the latent feature after the i^{th} downsampling layer, $\hat{F}_i$ the feature after ActNorm and the permuted channel layers, $\hat{F}_{ia}$ and $\hat{F}_{ib}$ the two features that have been split equally based on the channel level of $\hat{F}_i$. Then the output latent feature F_{i+1} from a NCCB can be expressed as:

$$\begin{aligned} \hat{F}_i &= \mathbf{p}(\mathbf{act}(F_i)) && (3.1) \\ &= \mathbf{W}^{\mathbf{p}}_{\theta} \otimes (\mathbf{S}^{\mathbf{act}}_{\theta} \times F_i + \mathbf{B}^{\mathbf{act}}_{\theta}) && (3.2) \end{aligned}$$

and

$$\begin{aligned} F_{i+1} &= \mathbf{coup}(\hat{F}_i) && (3.3) \\ &= \mathbf{coup}(\hat{F}_{ia}, \hat{F}_{ib}) && (3.4) \\ &= \mathbf{concat}(\hat{F}_{ia} \times \sigma(\mathbf{f}^{[\mathbf{1}]}_{\theta}(\hat{F}_{ib})) + \mathbf{f}^{[\mathbf{2}]}_{\theta}(\hat{F}_{ib}), \hat{F}_{ib}), && (3.5) \end{aligned}$$

where $\mathbf{W}^{\mathbf{p}}_{\theta}$ stands for the learnable weight matrix of the invertible 1x1 convolution. $\mathbf{S}^{\mathbf{act}}_{\theta}$ and $\mathbf{B}^{\mathbf{act}}_{\theta}$ are the learnable scale and shift parameters in ActNorm for a new mean and standard deviation. In Eq. 3.2, σ stands for the sigmoid operation and $\mathbf{f}^{[\mathbf{1}]}_{\theta}$ and $\mathbf{f}^{[\mathbf{2}]}_{\theta}$ represent two neural networks with parameters θ in each coupling block. $\otimes$ is the matrix multiplication and $\times$ is the multiply operation.

Figure 3.2b shows the reverse order of $NCCB$, it can be seen that we generate F_i using F_{i+1}. Denote $F_{(i+1)a}$ and $F_{(i+1)b}$ the two features that are equally split based on the channel level of F_{i+1}. Now we have:

$$\begin{aligned}\hat{F}_i &= \mathbf{coup^{-1}}(F_{i+1}) && (3.6)\\ &= \mathbf{coup^{-1}}(F_{(i+1)a}, F_{(i+1)b}) && (3.7)\\ &= \mathbf{concat}\left(\frac{F_{(i+1)a} - \mathbf{f}_\theta^{[\mathbf{2}]}(F_{(i+1)b})}{\sigma(\mathbf{f}_\theta^{[\mathbf{1}]}(F_{(i+1)b}))}, F_{(i+1)b}\right) && (3.8)\end{aligned}$$

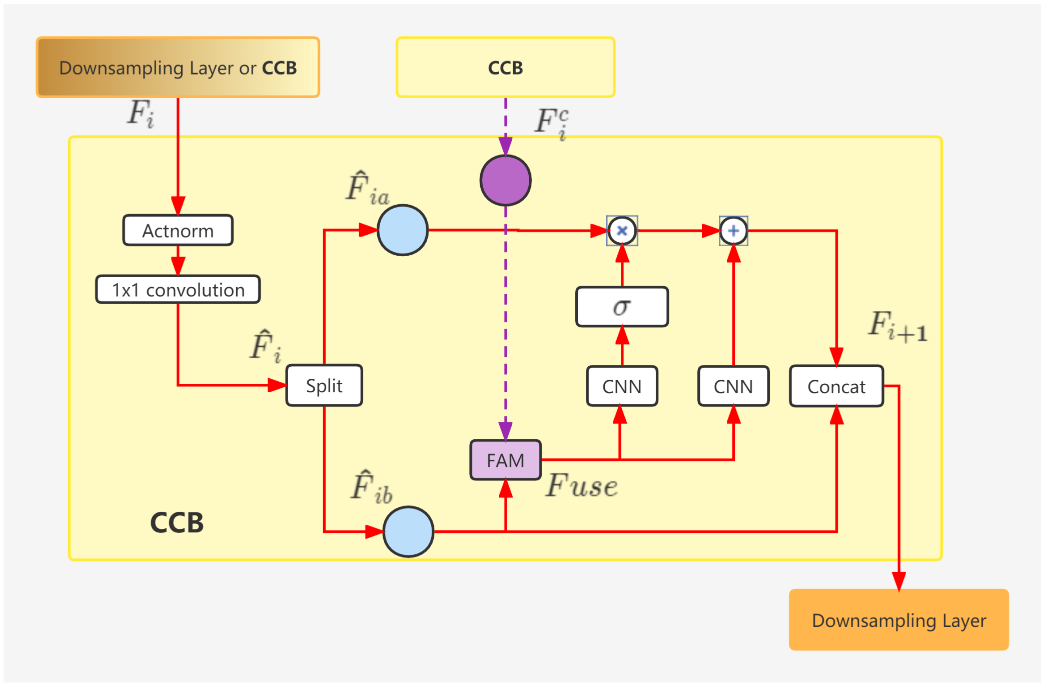
and

$$\begin{aligned}F_i &= \mathbf{act^{-1}}(\mathbf{p}^{-1}(\hat{F}_i)) && (3.9)\\ &= \mathbf{W}_\theta^{\mathbf{p^{-1}}} \otimes ((\hat{F}_i - \mathbf{B}_\theta^{\mathbf{act}})/\mathbf{S}_\theta^{\mathbf{act}}). && (3.10)\end{aligned}$$

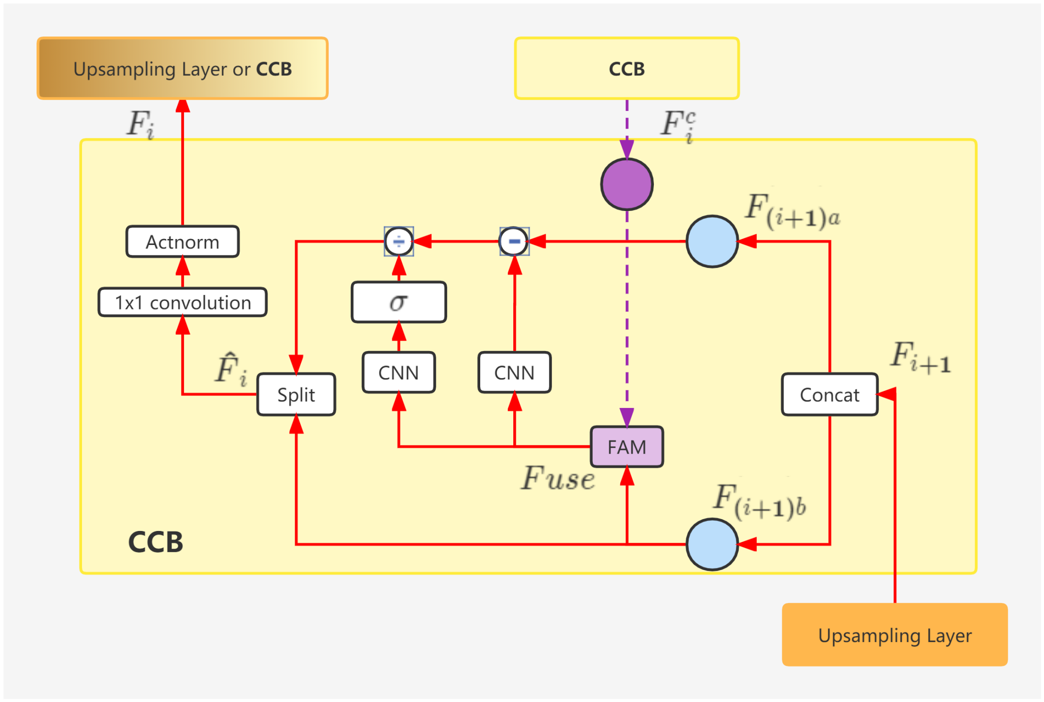
(2) The Conditional Coupling Block (CCB) is built upon the NCCB architecture and uses a Feature Aggregation Module (FAM) to fuse half of the latent feature $\hat{F}_{ib}$ with the color feature F_i^c, as shown in Figure 3.3. Following this, the fused latent feature is split into two sub-networks. Inside the CCB, the latent feature F_i comes from either the previous CCB block or the last downsampling layer. The i^{th} CCB recevies two inputs, the latent feature F_i from the previous CCB block or the last downsampling layer, and the color latent feature F_i^c from the corresponding CCB in the color encoder INN_c.

Here, the latent feature F_i received by the CCB represents two types of information. In the first case, as depicted by the red arrow in Figure 3.1, F_i represents the sketch feature F_i^s or F_{i+1}^s, which is used in the encoding process of INN_s and the decoding process of INN_c, respectively. The FAM in the CCB adaptively fuse the sketch feature with the color feature in this case. In the second case, illustrated by the purple dotted arrow in Figure 3.1, F_i^c represents the color feature, which has been used in the encoding process of INN_c. Since both inputs represent the color information, these latent features go through the FAM is just to ensure the invertibility of our normalizing flow. Then the output of the i_{th} CCB provides the color feature F_{i+1}^c to each corresponding CCB in the sketch encoder INN_s.

Inspired by the success of self-attention in transformers [52], our FAM also employs the attention mechanism. In the following, we provide an example of the FAM from



(a)



(b)

Figure 3.3: The architecture of conditional coupling block (CCB). (a) The encoding process of CCB where it takes a single input F_i conditioned with color information F_i^c and outputs feature F_{i+1}. (b) The decoding process of CCB where it takes a single input F_{i+1} conditioned with color information F_i^c and outputs feature F_i.

the CCB in the sketch Encoder INN_s, the details of which can be found in Figure 3.4. To begin with, half of the sketch latent feature $\hat{F}^s_{ib}$ and the color latent feature F^c_i are input into two independent multi-layer perceptrons (mlps), $\mathbf{mlp_s^{[1]}}$ and $\mathbf{mlp_c^{[1]}}$.

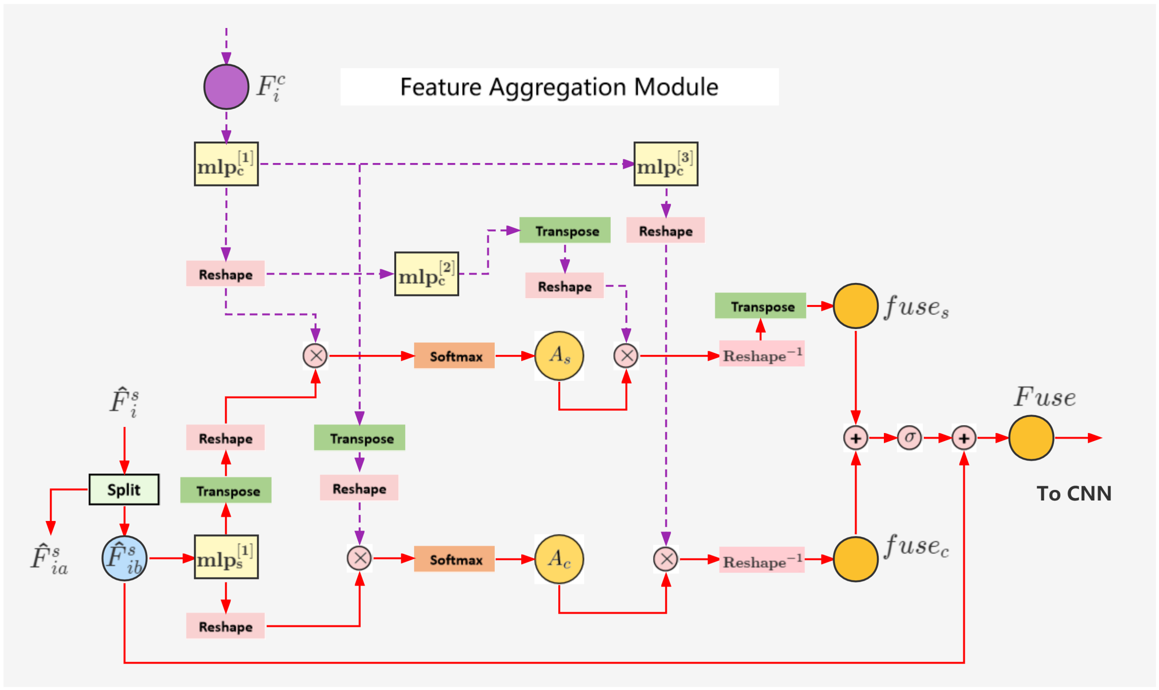


Figure 3.4: The proposed feature aggregation module. The purple and red lines indicate the workflow of the condition and input features, respectively. *Transpose* denotes the transpose operation between the channel and spatial dimensions, *Reshape* or **R** in an equation denotes the reshape operation which reduces the spatial dimension to 1, and $Reshape^{-1}$ or $\mathbf{R}^{-1}$ denotes the reverse. σ denotes the sigmoid function.

Denote the spatial-aware attention matrix as $A_s \in \mathbb{R}^{H\cdot W \times H \cdot W}$ and the channel-aware attention map as $A_c \in \mathbb{R}^{C\times C}$, the reshape operation $\mathbf{R} : \mathbb{R}^{C\times H\times W} \rightarrow \mathbb{R}^{C\times H\cdot W}$, and $\mathbf{R^{-1}}$ its inverse operation $\mathbb{R}^{C\times H\cdot W} \rightarrow \mathbb{R}^{C\times H\times W}$, then we have:

$$A_s = \mathbf{softmax}(\mathbf{R}(\mathbf{mlp_s^{[1]}}(\hat{F}^s_{ib}))^T \otimes \mathbf{R}(\mathbf{mlp_c^{[1]}}(F^c_i))) \tag{3.11}$$

and

$$A_c = \mathbf{softmax}(\mathbf{R}(\mathbf{mlp_s^{[1]}}(\hat{F}^s_{ib})) \otimes \mathbf{R}(\mathbf{mlp_c^{[1]}}(F^c_i))^T) \tag{3.12}$$

where the attention matrices A_s and A_c are calculated based on the channel and spatial information, respectively. Now we have the fused spatial-aware feature $fuse_s$

and the fused channel-awared features $fuse_c$, and the final fused feature $Fuse$ as:

$$fuse_s = \mathbf{R^{-1}}[A_s \otimes \mathbf{R}(\mathbf{mlp_c^{[2]}}(\mathbf{mlp_c^{[1]}}(F_i^c)))^T]^T, \tag{3.13}$$

$$fuse_c = \mathbf{R^{-1}}[A_c \otimes \mathbf{R}(\mathbf{mlp_c^{[3]}}(\mathbf{mlp_c^{[1]}}(F_i^c)))], \tag{3.14}$$

and

$$\begin{aligned} Fuse &= \mathbf{FAM}(\hat{F}^s_{ib}, F_i^c) && (3.15)\\ &= \hat{F}^s_{ib} \times [\sigma(fuse_s + fuse_c) + 1], && (3.16)\end{aligned}$$

where σ is the sigmoid function, $\mathbf{mlp_c^{[2]}}$ and $\mathbf{mlp_c^{[3]}}$ are another two independent mlps. Finally, the latent feature of the output sketch F^s_{i+1} from the i^{th} CCB is expressed as in Eq. 3.17:

$$F^s_{i+1} = \mathbf{concat}(\hat{F}^s_{ia} \times \sigma(\mathbf{f}_\theta^{[1]}(Fuse)) + \mathbf{f}_\theta^{[2]}(Fuse), \hat{F}^s_{ib}) \tag{3.17}$$

In the reverse order, the following equations demonstrate the calculation steps of our FAM and CCB in the color Encoder INN_c. Similar to Eq. 3.15 and Eq. 3.8, we have:

$$Fuse^{'} = \mathbf{FAM}(F^s_{(i+1)b}, F_i^c) \tag{3.18}$$

$$\hat{F}_i = \mathbf{concat}\left(\frac{F^s_{(i+1)a} - \mathbf{f}_\theta^{[2]}(Fuse^{'})}{\sigma(\mathbf{f}_\theta^{[1]}(Fuse^{'}))}, F^s_{(i+1)b}\right) \tag{3.19}$$

(3) For the choice of invertible downsamping or upsampling layers, according to Etman et al. [65], using pixel shuffle based or the wavelet transformation to downsample an image can create checkerboard artifacts. In order to generate a plausible color image, we follow the previous work [65] and use a learnable orthogonal downsampling operator as:

$$F_{i+1} = \mathbf{O_D}(x^i) = \mathbf{conv}(\mathbf{reshape}(\mathbf{exp}(\Theta - \Theta^T)), F_i) \tag{3.20}$$

For upsampling:

$$F_i = \mathbf{O_U}(x^i) = \mathbf{convT}(\mathbf{reshape}(\mathbf{exp}(\Theta - \Theta^T)), F_{i+1}) \tag{3.21}$$

where $F_{i+1} \in \mathbb{R}^{\frac{2N}{C} \times \frac{2N}{C}}$ is the image after downsampling and $F_i \in \mathbb{R}^{N \times N}$ is the image with a spatial size N before downsampling. $\theta \in \mathbb{R}^{C \times C}$ is a matrix parameterized by the Haar [66] kernels and C is the channel size after downsampling, for 2D data C is 4 [65]. **reshape** is defined as $\mathbb{R}^{C \times C} \rightarrow \mathbb{R}^{C \times 1 \times \frac{C}{2} \times \frac{C}{2}}$, which reorders the exponentials of matrices into a convolutional kernel [65]. **conv** and **convT** are convolution and transpose of convolution operations. $\mathbf{O_D}$ and $\mathbf{O_U}$ denote the orthogonal downsampling and upsampling functions, respectively.

3.2 Memory-efficient design

In order to make the computation of our model memory efficient, we designed our architecture to use fewer coupling blocks for shallow scale levels, where shallow scale levels imply that the latent features stay in a higher spatial size.

In our design, we use only one $NCCB$ between each downsampling layer, and then all the other CCBs are put after the last downsampling layer. Figure 3.5 illustrates the traditional design of the architecture in other flow-based methods. It can be seen that other methods [23, 24, 55, 59] use the same number of coupling blocks for each scale level. Particularly for very shallow information, more coupling blocks lead to more memory consumption. Thanks to the learnability of the orthogonal down or upsampling operation, the necessary spatial information can be adaptively converted into channel information, allowing the convolution networks $f_\theta^{[1]}$ and $f_\theta^{[2]}$ to more effectively use this information to estimate the affine coefficients.

We show the comparison between our model with other flow-based methods in Section 4.3. Additionally, an ablation study in Section 4.5.2 also shows that our model reduces computational cost that would have been spent on shallow information but still produces acceptable results.

3.3 Data preparation

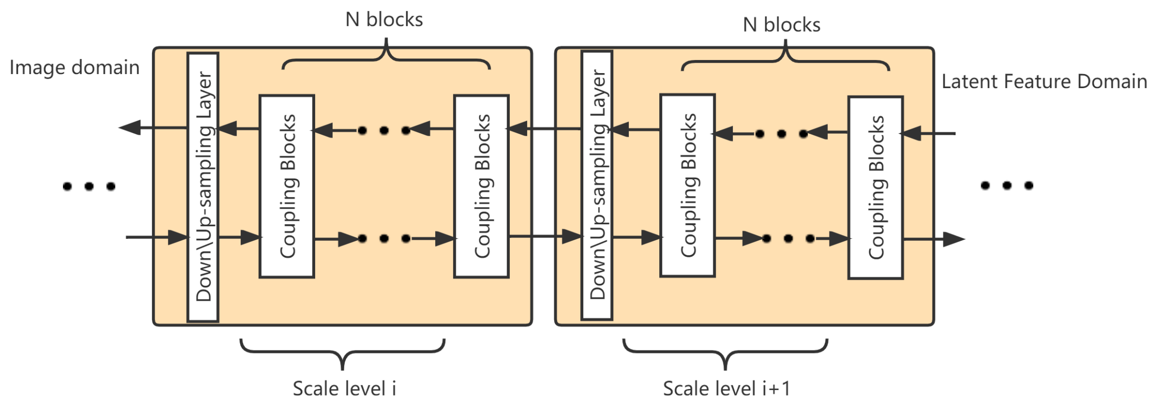


Figure 3.5: The traditional architecture design used in many flow-based models [23, 24, 55, 59]. For each scale level, they use N couplings blocks, while our method use only one coupling block for all scale levels except for the deepest one.

3.3.1 Dataset

Based on previous research in this field, we have chosen CelebAMask-HQ [67] as our dataset. This dataset contains 30K high-resolution (1024 x 1024) face images from public online sources, each paired with a segmentation mask of facial attributes. These attributes come from 19 classes that cover all facial components and accessories, such as skin, nose, eyes, ears, mouth, and more. Due to the high labeling costs associated with the larger size, the masks were manually annotated at a resolution of 512 x 512. However, they can be effortlessly upscaled from 512x512 to 1024x1024 using nearest-neighbor interpolation without causing noticeable artifacts.

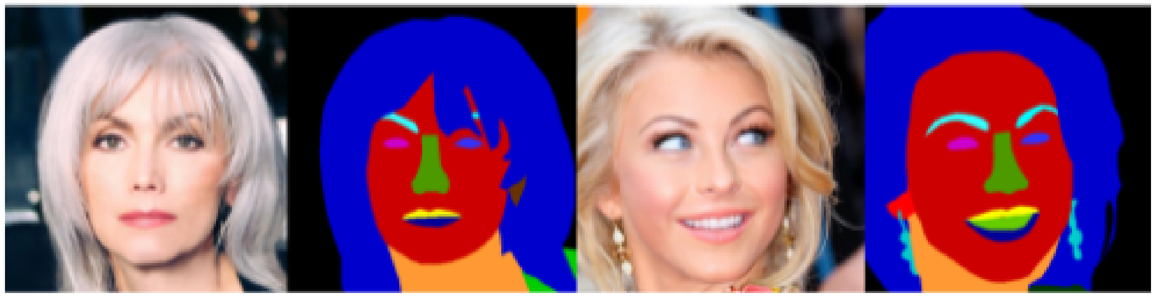


Figure 3.6: Examples from CelebAMask-HQ dataset [67].

We randomly split the data with a ratio of 4:3 into training and testing sets, and then resize all images from 1024 x 1024 into 256 x 256 to reduce the computational cost. Figure 3.6 shows same images from this dataset.

3.3.2 Sketch and mask generation

The original CelebAMask-HQ [67] includes color face images and their corresponding instance segmentation labels only. To create a face sketch from a color face image, we follow the method in previous works [21, 68], which uses the sketch filter in Photoshop to create an initial sketch and then a sketch simplification method [69] to simplify the sketch. Then we use the instance segmentation labels [67] to generate a binary mask of the face to remove the unnecessary background around the face, which helps the model focus more on the face region. Figure 3.7 gives an example on how we generate the face sketch.

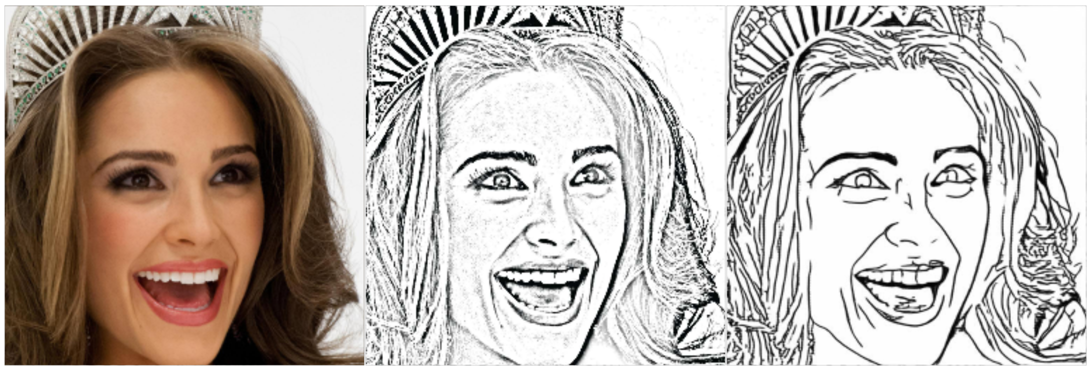


Figure 3.7: The left color image is from CelebAMask-HQ [67], the middle coarse sketch is obtained using the sketch filter in Photoshop, and the right sketch is generated by the sketch simplification method [69] from the middle sketch.

3.3.3 Data augmentation

Many existing conditional image translation methods [42, 44, 68, 70] use unsupervised cycle consistency training. However, these approaches usually include more encoding and decoding steps than typical supervised methods [54, 71–73], and require more training time. Therefore, in order to simplify the training process and to make the model adapt to different face images, inspired by the work of Lee, et al. [30], we adopt the self-augmented reference method to guide the colorization process.

Given both the original color face images and the corresponding face sketches, we apply random horizontally flip, rotation and center crop followed by an interpolation back to 256 x 256 resolution as the basic data augmentation to increase our sample diversity. Then, for the color images, we also randomly change their brightness, contrast, saturation and hue following the work of Lee, et al. [30], and then apply elastic transformations [74, 75] on the original color images, which are referred as the reference images. The purpose is to make the reference images geometrically independent from their corresponding sketches. The examples of the augmented images can be seen in Figure 3.1.

Elastic transformation, also known as elastic distortion, was first proposed by Patrice, et al. [76]. It randomly changes the shape of objects in images and creates a water-like effect by first generating random displacement fields. These fields are the changes of pixels' locations in a given image. For a pixel in an image at location (x, y), its corresponding displacement field is denoted as (δx, δy). Then, these fields are convolved with a Gaussian of standard deviation σ. We denote the final displacement field of each pixel as $(Conv(\sigma, \delta x)$, $Conv(\sigma, \delta y))$. Finally, the transformation moves each pixel to a new location at $(x + Conv(\sigma, \delta x)$, $y + Conv(\sigma, \delta y))$. An example of elastic transformation on the MNIST dataset [77] can be found in Figure 3.8.

During training, we apply the reconstruction loss that forces the model to conform with the original geometry but uses the color information from the reference image. Examples of self-augmented images can be found in the introduction to our overall model architecture, as depicted in Figure 3.1.

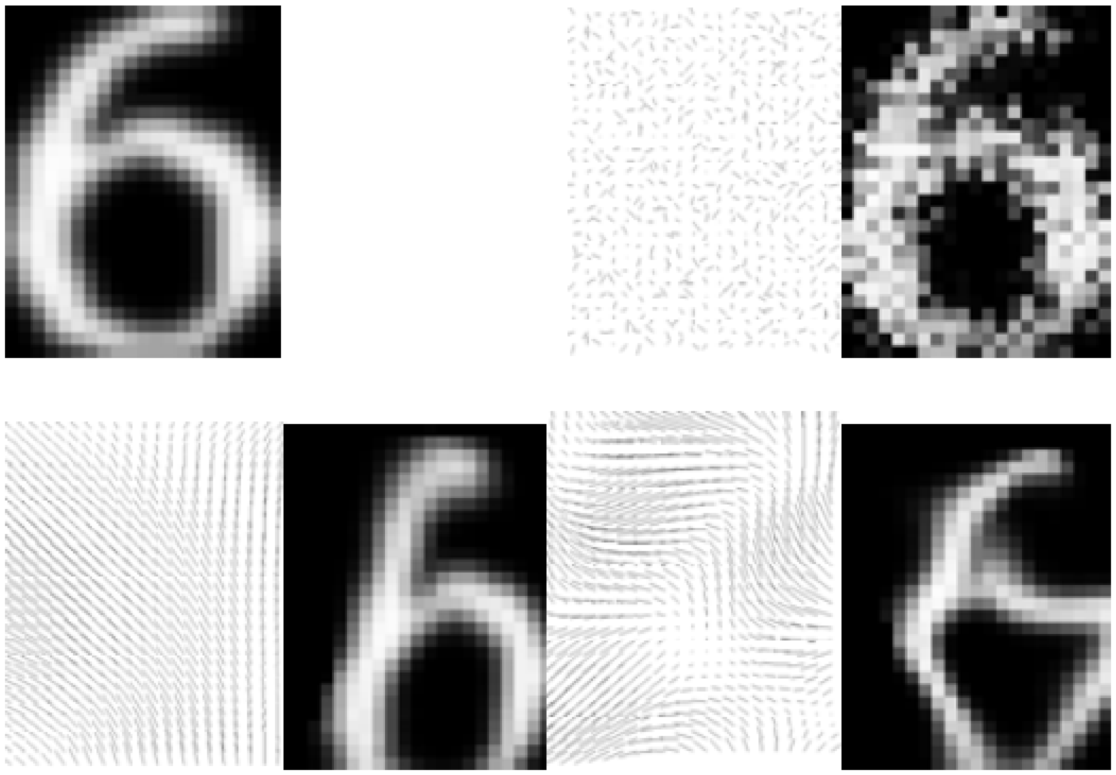


Figure 3.8: Top left: Original image. Right and bottom: Pairs of displacement fields and resulting images produced when different displacement fields are applied to the original image. Up right (σ=0.01), left bottom (σ=8), and right bottom (σ=4) [76].

3.4 Loss function

As introduced at the beginning of Section 3.1, I_{rec} represents the colorized image generated by our model, while I_{ref} is the paired color image from the dataset that has undergone elastic transformation and color permutation, as discussed in Section 3.3.3. Then, we define I_{gt} as the paired color image without any geometric transformation, ensuring that it shares a similar geometry with the sketch image.

To measure the accuracy of our colorized image, we use the L1-loss to calculate the absolute difference between the colorized image I_{rec} and the ground truth image I_{gt}. This loss is referred to as the Reconstruction Loss and is represented as follows:

$$L_{rec} = |I_{rec} - I_{gt}|, \tag{3.22}$$

To encourage our colorized images to be more natural and perceptually pleasing. we follow previous work by Johnson, et al. [78], which calculates square of the difference between the activation maps of the colorized image and of the ground truth image using a pretrained image classification network ($VGG16$). This loss is referred to as the Perceptual Loss [78] and is represented as:

$$L_{percp} = \sum_{i=1}^{|S|} |\phi_i(I_{rec}) - \phi_i(I_{gt})|^2 \ , \tag{3.23}$$

where ϕ_i denotes the activation map of the i_{th} layer of the pretrained $VGG16$ network, $|S|$ stands for the number of activation maps, where $S = \{3, 8, 15, 22, 29\}$ the set of layaer numbers, which has the same number of layers as in the previous work [78].

3.5 Implementation details

During training, the weight ratio of Reconstruction Loss to Perceptual Loss is set at 10:1. We use the Adam [79] optimizer with a batch size of 16, an initial learning rate of 0.0001 and a total of 100 training epochs. For every 5 epochs, the learning rate is reduced by a factor of 0.9.

In our model architecture, we set the total number of downsampling operations to 3, resulting in 3 $NCCB$s for each normalizing flow. Then, once the image has been transformed into the desired latent space, we put all the CCBs after the last downsampling layer. In practice, we use 8 CCBs for each INN model. As seen in Eq. 3.2, two independent neural networks are used in the coupling block to perform affine transformation. In practice, we choose a 3-layer residual block [80] followed by a 1x1 convolution layer for these networks. Each residual block contains a convolutional layer, an InstanceNorm layer [45] and a ReLU layer [81], and the last 1x1 convolutional layer is initialized with zeros [23]. The channel size of each residual network is the same as the input channel size, which means that for a shallow scale level, each layer has only a few parameters. Consequently, the majority of model parameters comes from our CCBs.

3.6 Summary

In this chapter, we introduced our sketch colorization method, which is based on two normalizing flows featuring exact image encoding and decoding, as well as with a feature aggregation module to effectively fuse color and sketch information. Our model is end-to-end for both training and testing, taking any color face image and unrelated face sketch as inputs and directly generating the colorized sketch without the need for intermediate preprocessing steps. By using learnable invertible downsampling layers, we can allocate fewer parameters to shallow level features, making our model more efficient in terms of computational resources, specifically memory, without sacrificing performance. This allows for faster training.

In the next chapter, we will carry out comprehensive qualitative and quantitative experiments to assess the performance and efficiency of our model.

Chapter 4

Experiments

In this chapter, we first introduce some state-of-the-art algorithms that serve as benchmarks for comparison with our method, as well as the evaluation metrics for quantitative assessments. We compare the performance of our proposed method with state-of-the-art methods using quantitative evaluations (Section 4.3) and qualitative comparisons (Section 4.4). Moreover, the results of an ablation study are discussed in Section 4.5 to analyze different settings of our method and to demonstrate the effectiveness of our design. Finally, we present the evaluation results from a user study in Section 4.6.

4.1 Compared methods

To demonstrate the performance of our proposed method, we select several algorithms that use a reference image as guidance to control the output of sketch to image colorization. We categorize these methods into CNN-based and normalizing flow-based methods.

For CNN based algorithms, the method proposed by Lee, et al. [30] is an end-to-end sketch to image colorization method which also employs the self-augmented reference training strategy. DeepFaceEditing [50] learns sketch to image translation through a cycle-consistency training. Liu, et al. [51] use a GAN to refine the synthesized image by employing a self-supervised Auto-Encoder.

For flow-based methods, An, et al. [24] apply style transfer by encoding and decoding different style domains using the same parameters. van der Ouderaa and Worrall [82] leverage an invertible mechanism in the high-dimensional feature space within a CycleGAN framework. Sorkhei, et al. [59] generate realistic images with the guidance from segmentation maps.

It is important to note that because DeepFaceEditing [50] does not have the training code, we can only use the provided pre-trained model to generate the colorized images. For all the other methods, we use their published codes and retrain their models on the CelebA [83] dataset with our created sketch images. During training, we use the default configuration settings of publicly available codes and limit the image size to 256 x 256.

4.2 Evaluation metrics

To quantitatively evaluate the performance of our method, we choose 4 evaluation metrics to assess our colorized images from different perspectives.

(1) To measure the overall quality of the colorized images, we use the **FID** [84] score to calculate the distance in a high-dimensional feature space between the colorized sketches and the real images, because FID score captures the disturbance level very well and has a better correlation with human judgment of visual quality. The terminology "disturbance level" means the degree of dissimilarity or difference between the real and generated images by a generative model. Following the implementation of Martin, et al. [84], we use the last pooling layer of a pretrained Inception-v3 model trained on the ImageNet dataset [85] as the coding layer to generate high-dimensional feature vectors. We denote the high-dimensional feature vectors of our colorized face sketch I_{rec} and reference color image I_{ref} as f_{rec} and f_{ref}, respectively. Furthermore, $\mu_{\mathbf{rec}}$ and $\mu_{\mathbf{ref}}$ represent the feature-wise means of f_{rec} and f_{ref}, while $\mathbf{C_{rec}}$ and $\mathbf{C_{ref}}$ are the covariance matrices for f_{rec} and f_{ref}, respectively. Therefore, the FID score can be calculated as in Eq. 4.1:

$$FID(I_{rec}, I_{ref}) = |\mu_{\mathbf{rec}} - \mu_{\mathbf{ref}}|^2 + Tr(\mathbf{C_{rec}} + \mathbf{C_{ref}} - 2(\mathbf{C_{rec}} \cdot \mathbf{C_{ref}})^{\frac{1}{2}}), \tag{4.1}$$

where Tr refers to the trace of a square matrix which is the sum of elements along the diagonal of the square matrix.

(2) Next, we follow the evaluation approach employed by An, et al. [24], we employ the **Structural Similarity Index Measure (SSIM)** [86] to measure the geometry similarity between the colorized sketches I_{rec} and the original real images I_{gt}. Unlike global metrics such as MSE, SSIM measures the similarity in local regions of images, which more closely aligns with human visual perception. Moreover, it extracts three key features—luminance, contrast, and structure—that are crucial for human perception of image quality, and the comparison between the two images is performed based on these features.

According to Wang, et al. [86], the luminance information of an image refers to the average values across all pixels. In this case, we represent the means of I_{rec} and I_{gt} using μ_{rec} and μ_{gt}, respectively. For contrast information, which is derived from the standard deviation of a given image, denoted as σ_{rec} and σ_{gt} for I_{rec} and I_{gt}, respectively. Then, the structural information is represented by the normalized values of the images. Lastly, SSIM is defined as a combination of all the above information and shown in Eq. 4.2:

$$SSIM(I_{rec}, I_{gt}) = \frac{(2\mu_{rec} \cdot \mu_{gt} + C_1)(2\sigma_{rec,gt} + C_2)}{(\mu_{rec}^2 + \mu_{gt}^2 + C_1)(\sigma_{rec}^2 + \sigma_{gt}^2 + C_2)} \tag{4.2}$$

where

$$\sigma_{rec,gt} = \frac{1}{N-1} \sum_{i=1}^{N} (I_{(rec)i} - \mu_{rec})(I_{(gt)i} - \mu_{gt}), \tag{4.3}$$

and N is the total number of pixel of an given image, $C1$ and $C2$ are very small constants to ensure stability when the denominator becomes zero.

(3) and (4) Finally, in accordance with Zhang et al. [53], we use **Content Relevance (CR)** to assess the semantic consistency between the colorized images I_{rec} and the original real images I_{gt}. Also, we use **Style Relevance (SR)** [53] to evaluate the

colorization accuracy between the colorized images I_{rec} and the corresponding reference image I_{ref}. Both CR and SR are computed using the cosine similarity of selected low-level features extracted from a pre-trained VGG16 network [38]. As suggested by Zhang et al. [53], we use the low-level feature of the 4^{th} and 9^{th} layers to compute the CR score, and of the 14^{th}, 23^{th}, and 32^{th} layers for high-level features to calculate the SR score. For the CR score, it is represented as:

$$CR(I_{rec}, I_{gt}) = \sum_{i=1}^{|S|} cos(\phi_i(I_{rec}), \phi_i(I_{gt})) \tag{4.4}$$

$$= \sum_{i=1}^{|S|} \frac{\phi_i(I_{rec}) \odot \phi_i(I_{gt})}{||\phi_i(I_{rec})|| \cdot ||\phi_i(I_{gt})||}, \tag{4.5}$$

where $\odot$ denotes the dot product operation and $||\phi_i(I_{rec})||$ is the L2 norm of a vector $\phi_i(I_{rec})$. ϕ_i denotes the activation map of the i_{th} layer of the pretrained $VGG16$ network, $|S|$ stands for the number of activation maps, where $S = \{4, 9\}$ denotes the set of layer numbers. For the SR score, similar to Eq. 4.4, it can be represented as in Eq. 4.6:

$$SR(I_{rec}, I_{ref}) = \sum_{i=1}^{|K|} cos(\phi_i(I_{rec}), \phi_i(I_{ref})), \tag{4.6}$$

where $|K|$ stands for the number of activation maps, and $K = \{14, 23, 32\}$ the set of layer numbers.

4.3 Quantitative comparison

Table 4.1 presents the quantitative comparison results between our method and the state-of-the-art approaches mentioned in Section 4.1. It can be seen that our method is competitive across all metrics when compared with other methods. In the first column, the FID score, the method proposed by Liu et al. [51] achieves the best result, but struggles to maintain sketch geometry, as evidenced by its significantly lower SSIM and CR scores. In contrast, our method achieves a close second-best FID score of 41.8, only slightly behind Liu et al. [51]. For the SR score, FullGlow [59]

Methods	FID $\downarrow$	SSIM $\uparrow$	SR $\uparrow$	CR $\uparrow$
RevGAN[82]	206.3	0.590	0.619	0.502
ArtFlow[24]	121.5	0.578	0.634	0.544
FullGlow[59]	59.8	0.621	**0.742**	0.648
DeepFaceEditing*[50]	47.7	0.703	0.665	0.738
Self-Supervised[51]	**41.3**	0.705	0.661	0.747
ReferenceBased[30]	44.3	0.711	0.668	0.741
Ours	41.8	**0.773**	0.684	**0.807**

Table 4.1: Quantitative comparison to both flow-based (Blue) and CNN-based methods (Red). For FID score, the lower the better, for SSIM, SR and CR score, the higher the better.

takes the top position. However, its inferior FID, SSIM, and CR scores indicate that it has difficulty to accurately applying color information to the face sketch in terms of perceptual quality from a human perspective and sketch geometry preservation.

It is clear that the CNN-based methods generally outperform the normalizing flow based methods, achieving superior scores across all four metrics. However, when considering both color accuracy and geometry consistency, our normalizing flow-based method achieves the second-best SR and FID scores, as well as the highest CR and SSIM scores. This highlights the effectiveness of our approach, which uses normalizing flow for sketch colorization, and indicates that our method can apply colors to a sketch more accurately while preserving sketch geometry.

In Table 4.2, we compare the training efficiency of our baseline model with other normalizing flow based models [24, 59, 82] using three metrics: the number of model parameters, GPU memory usage, and Giga Multiply-Accumulate operations (GMACs). GMACs is a metric used to measure the computational complexity of neural networks and provides an estimate of the computational resources required for the network, where a lower GMACs value indicates a more computationally efficient model. As shown in Table 4.2, our model achieves the lowest GMACs value, leading to the most

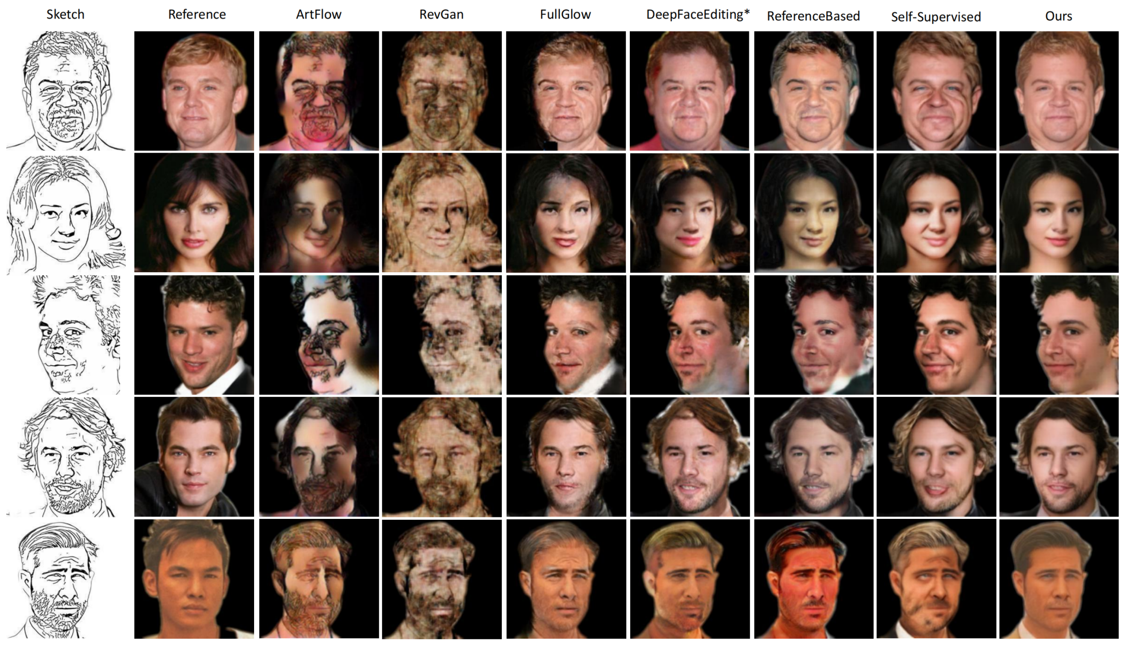
Arch/ loss	params	GPU memory usage	GMACs
ArtFlow[24]	**6.5m**	52G	57.7
FullGlow[59]	82.3m	13G	157.7
RevGan[82]	6.8m	15G	140.6
Ours	17.1m	**12G**	**17.5**

Table 4.2: A comparison between our model with other flow-based methods in the number of parameters, GPU memory usage and GMACs. All the trainings follow the default settings.

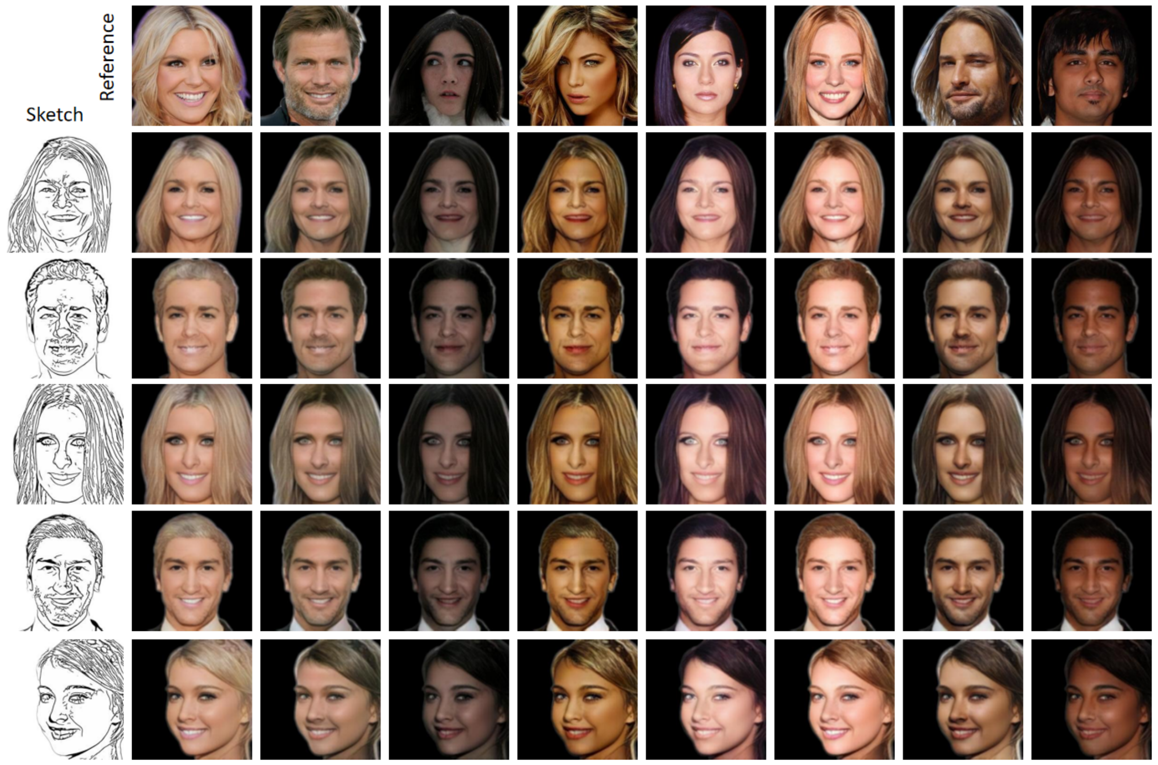
efficient training. On the other hand, other models [24, 82] use fewer parameters but consume more GPU memory and require longer training times.

4.4 Quanlitative comparison

Figure 4.1a shows the visual comparison between our method and other methods. Start from the third column, ArtFlow [24], which simply incorporates a style transfer module and a training strategy with an flow-based model, cannot properly add colors to a given sketch image. RevGAN [82] fails to colorize the sketches. While FullGlow [59] and the pretrained DeepFaceEditing [50] present a realistic color quality on their colorized images, they cannot ensure that the color follows the geometry of the sketch as precisely as possible. For example, in the second row, not only do the colorized images not resemble the input sketches, but also a significant part of the girl's face is in shadow, which is not realistic. Next, ReferenceBased [30] and Self-Supervised [51] can follow the geometry of the sketch in their colorized images, but the colors from ReferenceBased [30] shown in the seventh column deviate from that of the reference images shown in the second column, and Self-Supervised [51] overemphasizes the edges. In contrast, the colorized images of our method shown in the last column demonstrate a good balance between geometry consistency (input sketch) and color accuracy (reference image). In Figure 4.1b, we present more colorization examples by our approach. It can be seen that our method accurately colorize a sketch with



(a)



(b)

Figure 4.1: (a) Qualitative comparison with other baseline methods, The first column is the input sketch image and the second column is the corresponding reference image. The last column is the colorized results using our method. (b) Qualitative comparison of our method that each sketch image (in the first column) takes 8 colorful images (in the second column) as exemplars.

respect to different color images as references. We show more diverse examples in Appendix A.

4.5 Ablation study

We perform ablation studies to cover three aspects: the proposed FAM module, the model architecture, and the loss functions.

4.5.1 Analysis of the Feature Aggregation Module

To demonstrate the effectiveness of the FAM, we conduct an ablation study by evaluating different feature aggregation methods as alternatives to our FAM. Along with our baseline model, we examine three additional variations: (a) a model without FAMs, (b) a model using concatenation operation to fuse features, as suggested in [59], and (c) a model that employs AdaIN for transferring color information to the sketch, as described in [24]. In every ablated model, the rest of the components remain the same as in our baseline model. This consistency allows us to draw reliable and insightful conclusions about the impact of various feature aggregation methods on our model's performance.

The visual comparsion results are shown in Figure 4.2. It is clear that, without any feature aggregation strategy (b), the output image appears to be lacking in colors. Additionally, the other two methods (c) and (d) tend to produce color images with poor alignment to either the color reference image or the sketch, especially when there are significant geometric differences between them. For example, in the first and last columns, both (c) and (d) appear to ignore the sketch geometry, applying colors directly onto the sketch without any adaptation to its geometric structure. The quantitative results in Table 4.3 further support this observation, as both (c) and (d) show poor SSIM and CR scores, indicating their failures to preserve the appropriate geometry. In contrast, our proposed FAM (a) consistently offers superior colorized results, outperforming the other variations and validating the effectiveness of our

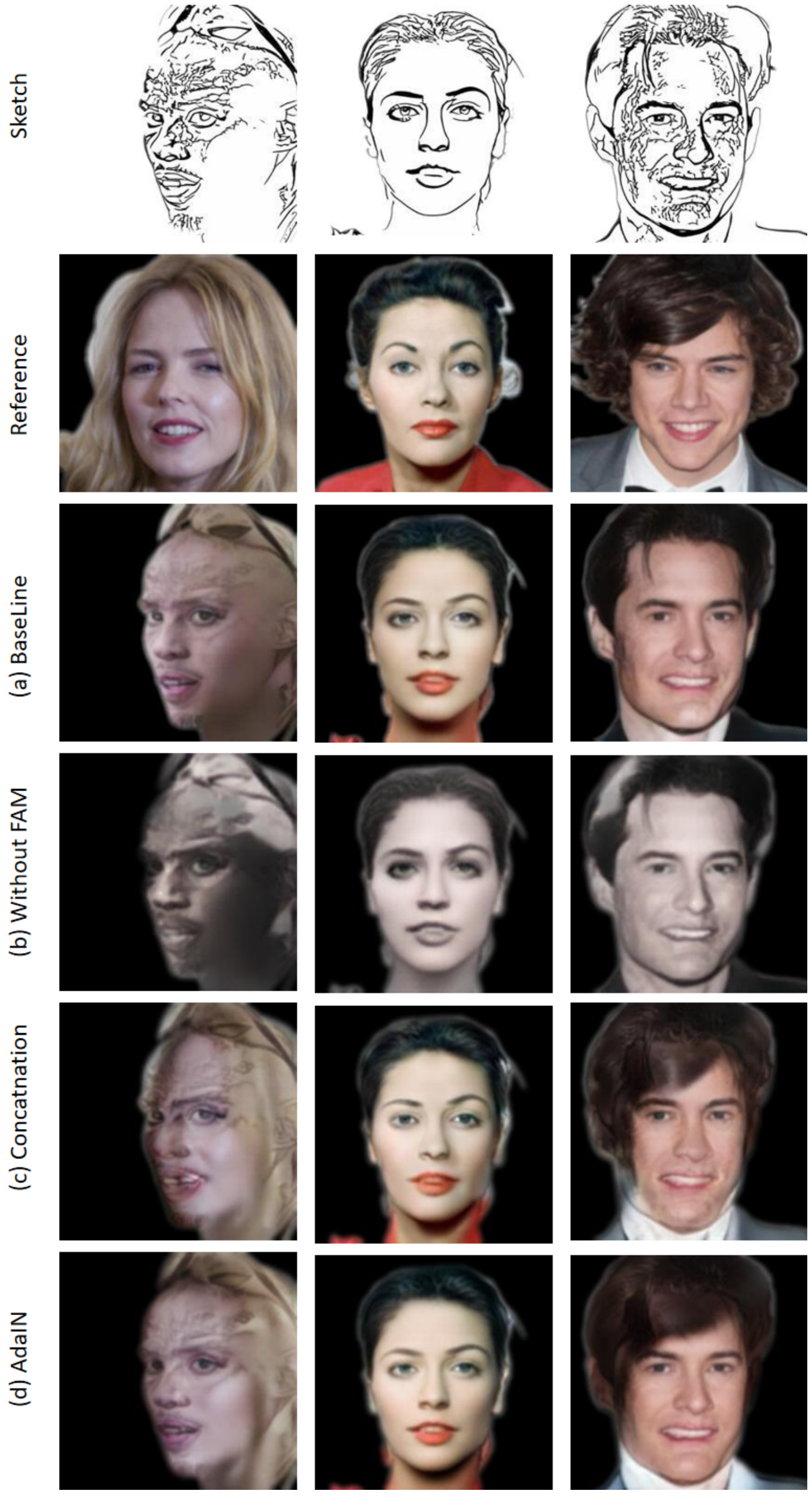


Figure 4.2: An ablation study with variants that substitute the proposed Feature Aggregation Module with different methods.

	FID $\downarrow$	SSIM $\uparrow$	SR $\uparrow$	CR $\uparrow$
without FAM	62.0	0.764	0.561	0.759
concatenation	42.3	0.668	0.702	0.709
AdaIN	**39.1**	0.699	**0.705**	0.752
Baseline	41.8	**0.773**	0.684	**0.807**

Table 4.3: Quantitative comparison on our method with different Feature Aggregation Modules

approach.

4.5.2 Analysis of the model architecture

To demonstrate the effectiveness of our model in reducing computational resources, we carry out an ablation study that compares our baseline model with various alternative models. These alternatives either use more coupling blocks for shallow features or downsample images with a different number of times. Likewise, other components maintain identical to our baseline model. As mentioned in Section 3.5, our baseline model incorporates 3 downsampling layers with a total of 8 $CCBs$ and includes 1 $NCCB$ between every two downsampling layers. For the variant models, we either reduce the image size by a factor of two or four, and use a combination of the number of $NCCB$s and downsampling layers to denote the names of different models. For instance, $NCCB(2, 1)$ refers to the variant that downsamples the image twice and has one $NCCB$ between every two downsampling layers. Another example, $NCCB(4, 8)$ represents a variant model that features 4 downsampling layers and 8 $NCCBs$ in between every two of downsampling layers. Additionally, we also compare variants that use the same number of coupling blocks at each scale level, which in our case is 8.

We present the qualitative results in Figure 4.3. It can be seen that the variant models (a) and (b) maintain a good geometry but perform worse in capturing color information, resulting in lower overall scores shown in Table 4.4. We believe that these

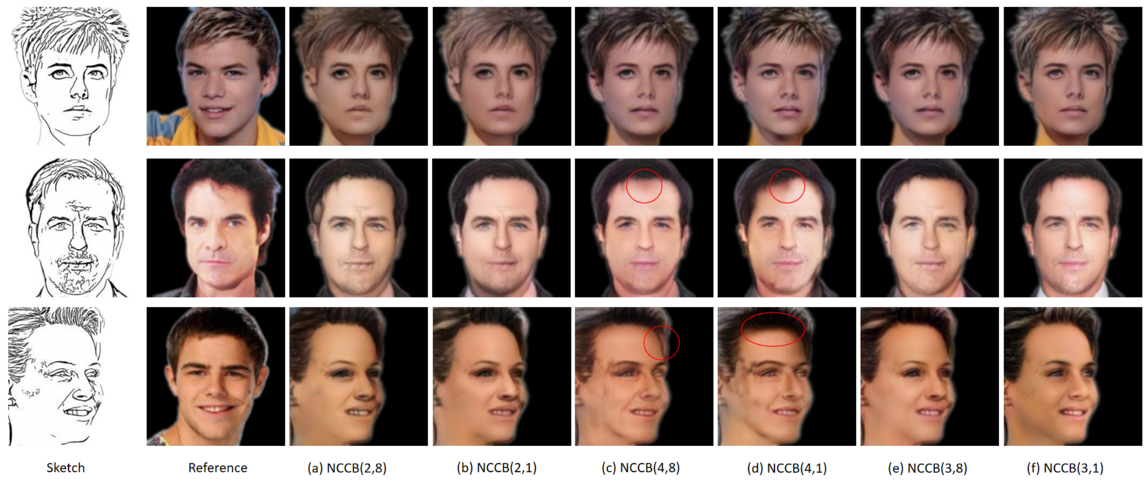


Figure 4.3: An ablation study using a different number of downsampling layers and $NCCB$s in our desgin.

variants are too small to get good results, as can be seen in Table 4.5. Although these variants have very few parameters, they consume a large amount of GPU memory. This is due to the use of a fewer number of downsampling layers on an image, which requires higher spatial resolution information and demands more GPU memory for computation.

For the variant models (c), (d), (e) and (f) with more downsampling layers, they have more parameters but use less memory. This is due to the lossless transfer of spatial information to the channel dimension, which indicates that a larger kernel size used in the convolution layers of the coupling blocks. However, more parameters does not guarantee better performance. Although variant models (c) and (d) give better FID and SR scores than our baseline, the color bleeding effect is more apparent when 4 downsampling layers are used. For example, in the fifth and sixth column in Figure 4.3, the forehead regions inside the red circle of the man and the woman have the black color of the hair region leaked to the face region.

Table 4.5 also demonstrates that when having more coupling blocks for each scale level does not necessarily lead to improvement in performance. With fewer parameters, although the training efficiency is improved, more GPU memory is used and

Arch	FID	SSIM	SR	CR
NCCB(2,1)	44.9	0.759	0.668	0.783
Ours NCCB(3,1)	41.8	**0.773**	0.684	**0.807**
NCCB(4,1)	**40.9**	0.713	**0.695**	0.752
NCCB(2,8)	43.8	0.745	0.653	0.781
NCCB(3,8)	42.7	0.761	0.679	0.774
NCCB(4,8)	41.2	0.720	0.689	0.748

Table 4.4: Quantitative comparison of our method with the variants has different number of downsampling layers and $NCCB$s.

Arch	#params	GPU memory usage	GMACs
NCCB(2,1)	**0.8m**	46G	**1.7**
Ours NCCB(3,1)	17.1m	**12G**	17.5
NCCB(4,1)	138.3m	13G	37.6
NCCB(2,8)	1.2m	57G	2.1
NCCB(3,8)	25.2m	21G	19.4
NCCB(4,8)	310.0m	23G	45.2

Table 4.5: The number of parameters and the required GPU memory for training our variant models. All the experiments use the same setting from Section 3.5

the colorization quality is lower. This means that our memory-efficient design can achieve acceptable results with a modest number of model parameters and a low memory footprint.

The recent success of the GPT-4 model [87] has demonstrated remarkable performance, attributed to its use of over 10 trillion parameters. While having more parameters often leads to better performance, it is essential to consider that not every research group or department possesses the computational resources or time required to work with such massive models. Consequently, it is equally important to develop models that achieve strong performance with as few parameters as possible.

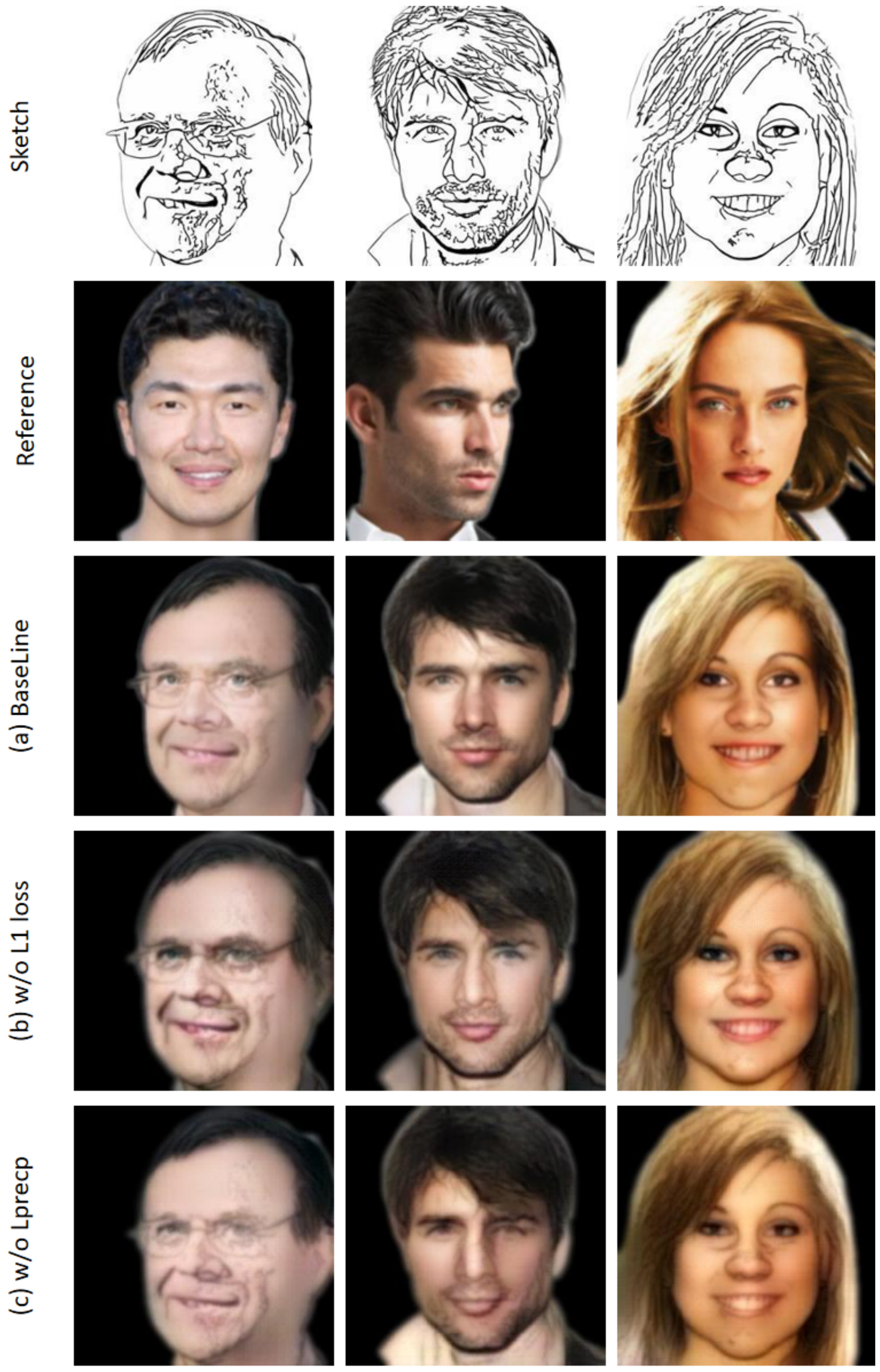


Figure 4.4: Ablation study of the loss terms.

loss	FID	SSIM	SR	CR
Ours + w/o Lpercp	53.4	0.745	0.681	0.755
Ours + w/o L1	49.3	0.755	0.763	0.784
Ours	**41.8**	**0.773**	**0.684**	**0.807**

Table 4.6: Analysis of using loss terms.

4.5.3 Analysis of loss functions

We modify the loss function to investigate the influence of each loss term on the final results. In Figure 4.4, we observe that when solely using the L1 loss (c), the resulting colorized sketches appear more pale in comparison to our baseline model (a). On the other hand, if we keep only the perceptual loss (b), the colorized sketches visually resemble our baseline but exhibit a blurry appearance. The quantitative outcomes presented in Table 4.6 further substantiate that incorporating all loss terms yields the highest overall quality and colorization accuracy. This analysis highlights the importance of combining various loss terms to achieve optimal results in our model.

4.6 User study

Due to the lack of ground truth for our colorized sketches, the evaluation metrics mentioned earlier serve as indirect measures of the colorized sketch quality. To further complement these metrics, we carry out a user study involving human subjects to evaluate the performance of our model compared to state-of-the-art methods. This user study aims to qualitatively demonstrate that our proposed model can effectively maintain accurate geometry while capturing precise colors from the reference image. There are two main advantages of conducting user study: (1) A user study allows for a subjective assessment of the colorization results, considering human perception and aesthetic preferences that may not be fully captured by quantitative metrics. (2) Involving human participants in a user study provides real-world validation of the model's effectiveness in generating visually appealing and geometrically faithful colorized images to the sketch, ensuring that the results are practical from human's perspective.

Incorporating a user study helps us gain a better understanding of our model's performance. In the following, we describe the design of our user study.

4.6.1 User study design

When inviting participants for our user study, it is important to take some practical factors into account. Firstly, we avoided choosing people we know to participate in the survey, as they are more likely to recognize which method is ours when viewing the comparison images, and they tend to give higher scores to our method. This could lead to biased feedback. To address this concern, we followed the approach used in previous works [50] and distributed the survey online to strangers.

Secondly, we also need to think about people's willingness to participate in the survey and find effective ways to encourage more strangers to join in the survey. To address this concern, we chose a reputable platform called 'Wenjuanxing', a platform providing functions equivalent to Amazon Mechanical Turk. This platform is specifically designed for creating, distributing, and managing surveys, and it also maintains participants' anonymity and data privacy. By ensuring that no personally identifiable information is collected and that responses are kept confidential, participants may feel more comfortable providing honest feedback.

Thirdly, it's important to make sure that the survey is easy to access, user-friendly, and time-efficient for both participants and researchers. To accomplish this, we used clear and simple language in the survey, avoiding overly long or complex questions. By keeping participants engaged and motivated to provide their feedback, we can also save time during data analysis.

Therefore, we conduct our user study using a questionnaire with two different questions. To create a comprehensive set of comparison examples for this study, we randomly select 30 sketch images and color reference images from the test dataset.

To guarantee a thorough evaluation, we divide these examples into two equal halves. The first half, which has 15 examples, is dedicated to the first question on evaluating the geometry preservation in the colorized images. In particular, this question asks how well the colorized images maintain the geometric details and structure of the original sketches. This assessment helps in determining the model's effectiveness in preserving the sketch geometry of the colorized image.

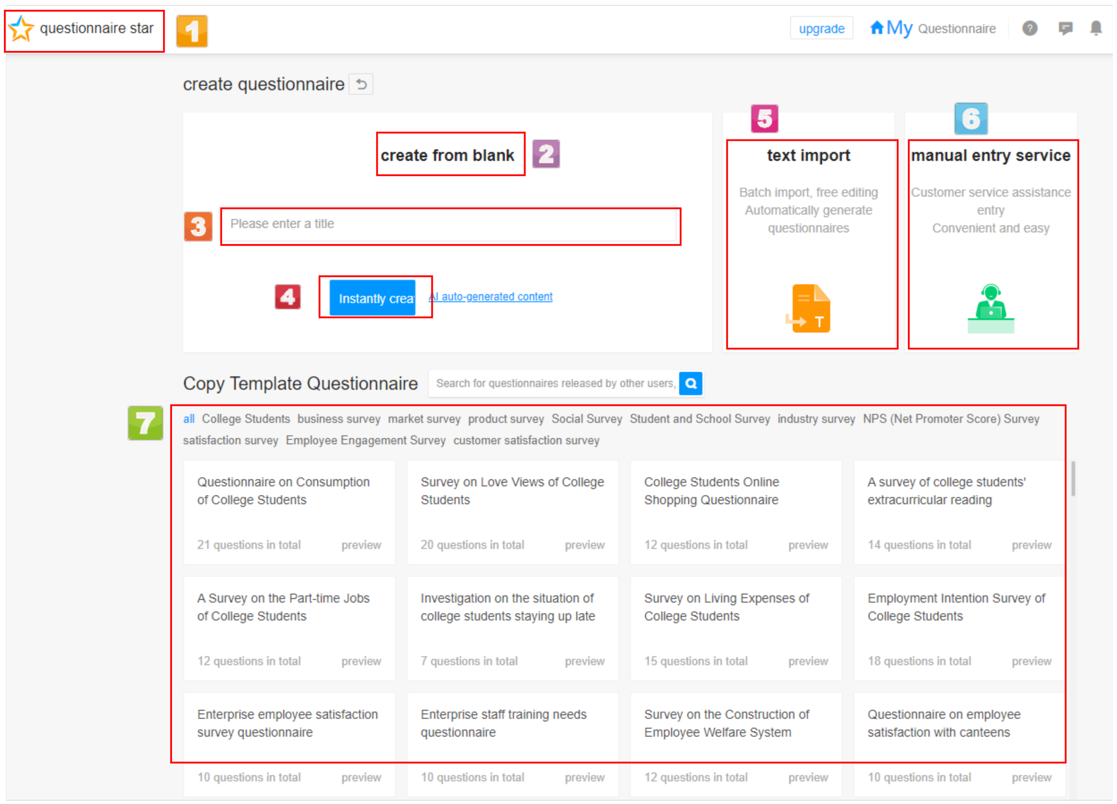
The second half, with another 15 examples, is set aside for the second question, which evaluates the quality of colorization and its accuracy with respect to the reference image. This evaluation considers factors such as color fidelity, consistency, and the overall aesthetic appeal of the colorized images. Assessing colorization quality helps determine if the model can produce visually pleasing and accurate colorized results that align with the reference image.

In our user study, for each question, we use a 5-point rating scale that participants apply to evaluate the performance of each method according to their judgment. The examples of these questions can be seen in the subsequent subsection 4.6.3, where we also give a brief introduction of the website we used for gathering and managing the questionnaires.

4.6.2 User study platform

In this subsection, we provide a visual guide to show how we make questionnaires on the 'Wenjuanxing' platform, including examples of the two types of questions mentioned earlier. As the platform is primarily in Chinese, we translate the whole page to English and and explain key content inside each red rectangle in each image's captions for easier understanding.

1: 'Wenjuanxing' is a user-friendly platform that simplifies the survey creation process, making it more convenient to design and carry out the questionnaire. The website's main page, displaying its features and offerings, can be seen in Figure 4.5:


questionnaire star 1

upgrade My Questionnaire

create questionnaire

create from blank 2

3 Please enter a title

4 Instantly crea AI auto-generated content

5 text import

Batch import, free editing
Automatically generate questionnaires

6 manual entry service

Customer service assistance entry
Convenient and easy

Copy Template Questionnaire Search for questionnaires released by other users

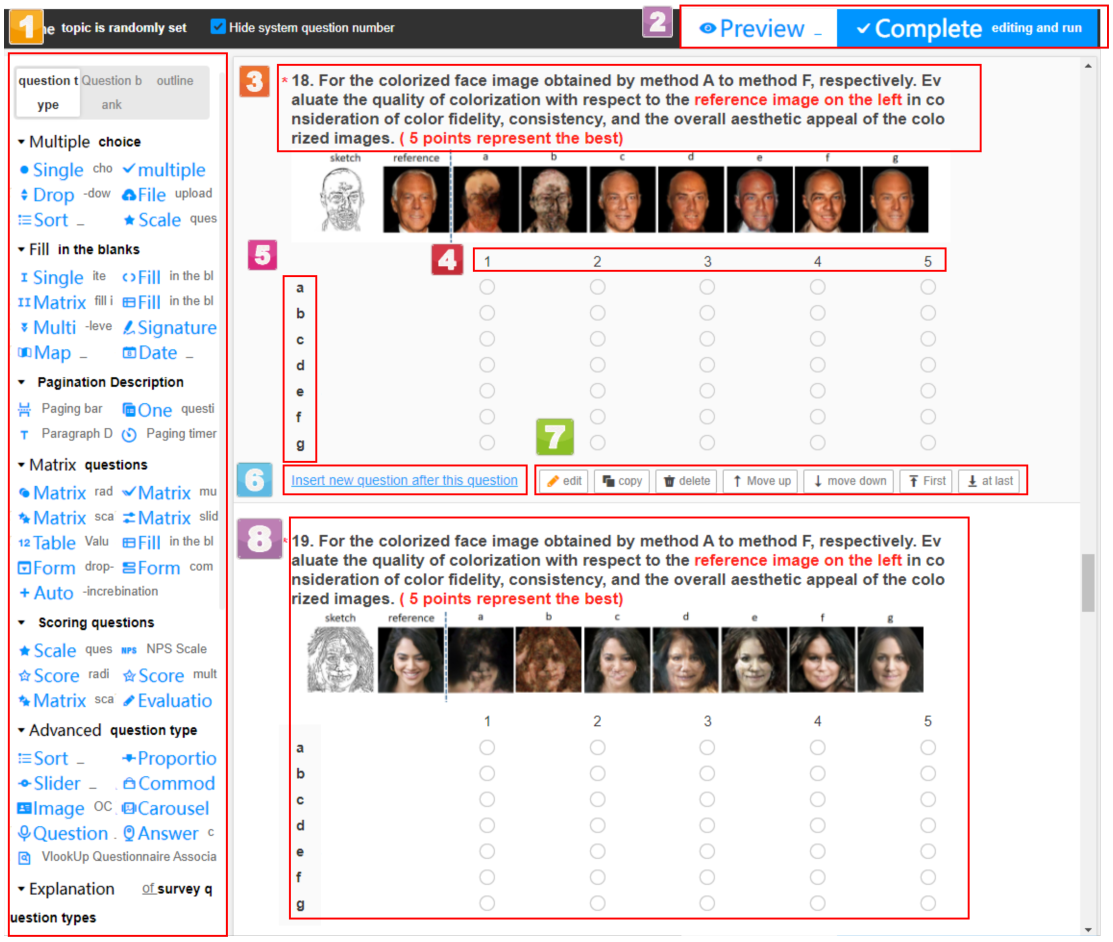
7 all College Students business survey market survey product survey Social Survey Student and School Survey industry survey NPS (Net Promoter Score) Survey satisfaction survey Employee Engagement Survey customer satisfaction survey

Questionnaire on Consumption of College Students 21 questions in total preview	Survey on Love Views of College Students 20 questions in total preview	College Students Online Shopping Questionnaire 12 questions in total preview	A survey of college students' extracurricular reading 14 questions in total preview
A Survey on the Part-time Jobs of College Students 12 questions in total preview	Investigation on the situation of college students staying up late 7 questions in total preview	Survey on Living Expenses of College Students 15 questions in total preview	Employment Intention Survey of College Students 18 questions in total preview
Enterprise employee satisfaction survey questionnaire 10 questions in total preview	Enterprise staff training needs questionnaire 10 questions in total preview	Survey on the Construction of Employee Welfare System 12 questions in total preview	Questionnaire on employee satisfaction with canteens 10 questions in total preview

Figure 4.5: An overview of the main page of the survey platform 'Wenjuanxing'. **Rectangle 1** displays the platform's name. **Rectangle 2** indicates where users can create a new survey from scratch. **Rectangle 3** is the area for inputting survey titles. **Rectangle 4** shows the 'create' button for initiating a new survey. **Rectangle 5** highlights the option to generate a survey from text input. **Rectangle 6** reveals the support for creating a survey through customer service assisted by the platform. **Rectangle 7** lists some templates of various surveys for user convenience.

2: By clicking the 'create' button, we are redirected to the page where we can edit the contents of our survey. As depicted in Figure 4.6, the editing interface showcases an example question and provides an easy-to-use environment for customizing the survey's content.

3: Figure 4.7 also presents an example question addressing the preservation of geometric information in colorized face sketches. The question evaluates whether the geometry is clearly identifiable and if the colors are applied both strictly and reasonably, adhering to the sketch's original geometric structure.



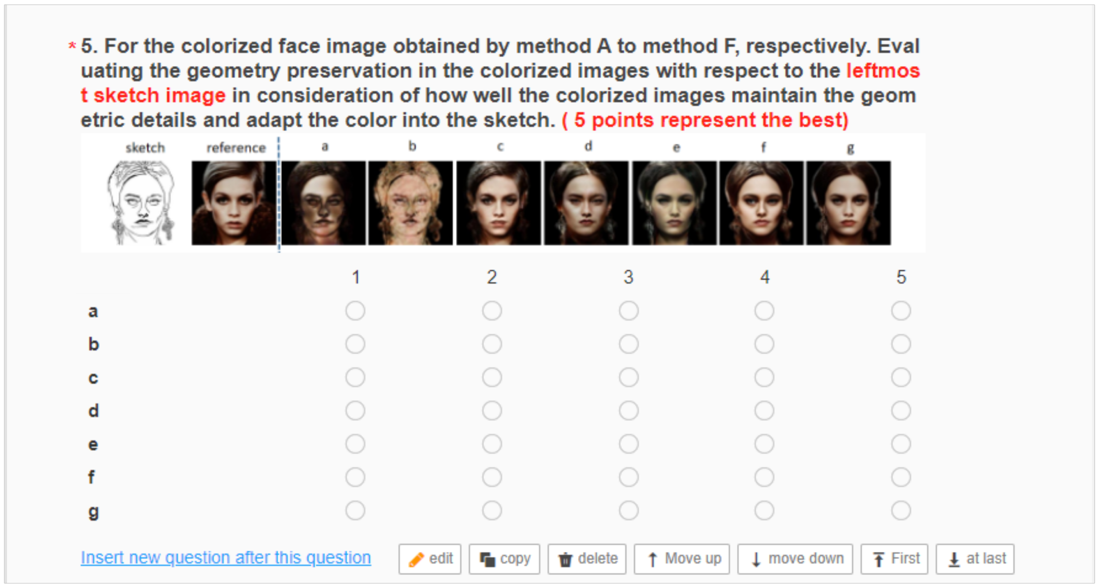
* 18. For the colorized face image obtained by method A to method F, respectively. Evaluate the quality of colorization with respect to the reference image on the left in consideration of color fidelity, consistency, and the overall aesthetic appeal of the colorized images. (5 points represent the best)

* 19. For the colorized face image obtained by method A to method F, respectively. Evaluate the quality of colorization with respect to the reference image on the left in consideration of color fidelity, consistency, and the overall aesthetic appeal of the colorized images. (5 points represent the best)

Figure 4.6: An overview of the editing page on the survey platform 'Wenjuanxing', which includes our questions. **Rectangle 1** represents the various types of questions that can be created. **Rectangle 2** shows the buttons that stand for "Preview" and "Finish Editing." **Rectangle 3** displays the content of each question, stating, "For the colorized face images obtained by methods A through F, please rate their color accuracy in comparison to the reference image on the left. **Rectangle 4** presents the grades assigned to each option. **Rectangle 5** lists the 7 comparison methods along with their corresponding marking options. **Rectangle 6** shows the button to add question behind the current one **Rectangle 7** contains the buttons to edit the questions. From left to right, they are: Edit, Copy, Delete, Move up, Move down, To the front, To the back. **Rectangle 8** shows the next question.

4: Once finished editing the survey, we can start to send the survey online. This platform provides a wide range of sharing options, as illustrated in Figure 4.8

5: After participants finish the survey, the platform offers various data analysis options. In our case, as shown in Figure 4.9, we counted five distinct grades


* 5. For the colorized face image obtained by method A to method F, respectively. Evaluating the geometry preservation in the colorized images with respect to the leftmost sketch image in consideration of how well the colorized images maintain the geometric details and adapt the color into the sketch. (5 points represent the best)

	1	2	3	4	5
a	○	○	○	○	○
b	○	○	○	○	○
c	○	○	○	○	○
d	○	○	○	○	○
e	○	○	○	○	○
f	○	○	○	○	○
g	○	○	○	○	○

Figure 4.7: An example of the question that evaluate the geometry perservation on the colorized face sketch. The layout is similar to the question in 4.6

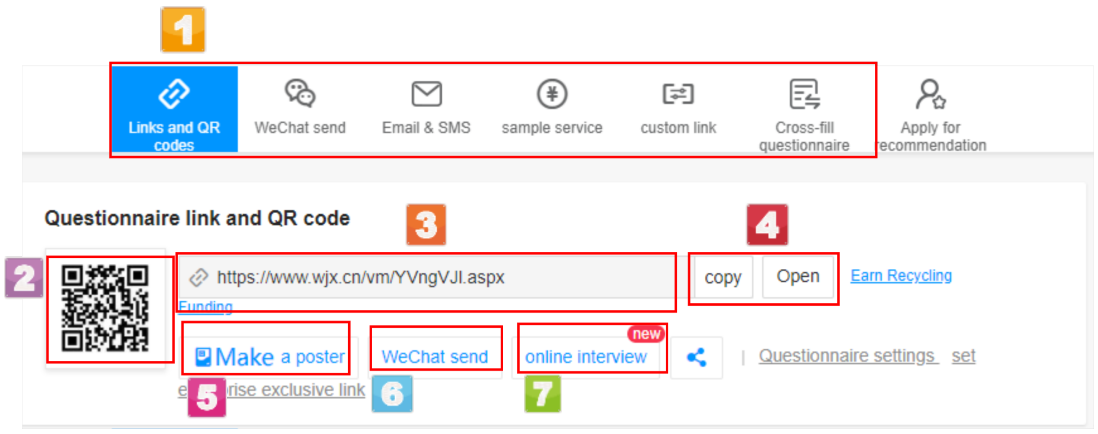
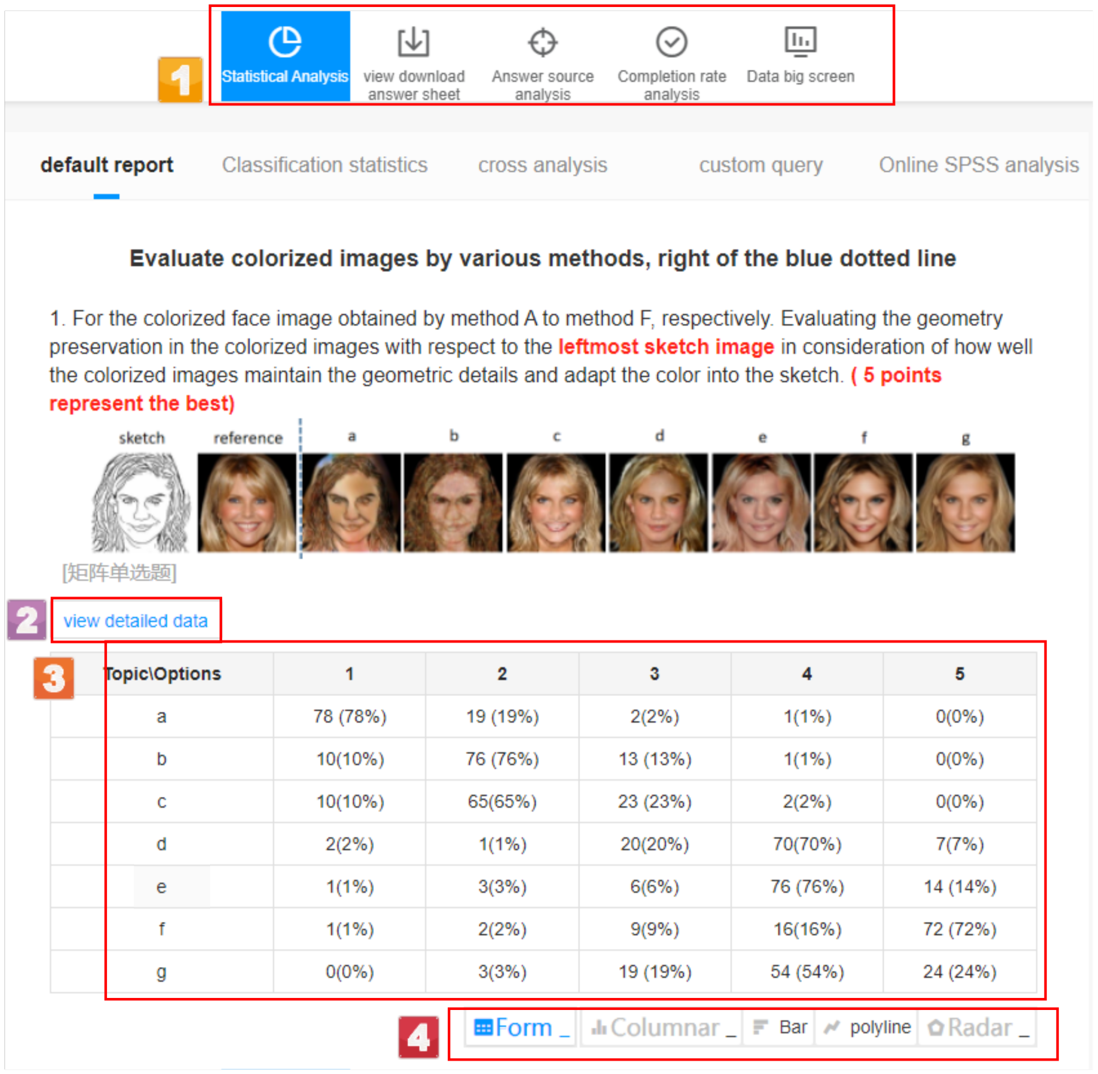


Figure 4.8: **Rectangle 1** displays various sharing options for our survey. From left to right, they include sharing via link and QR code (the current page), sharing through WeChat, sharing via email or text message, utilizing paid services for survey distribution, customizing the sharing link, and posting the survey to the community. **Rectangle 2** contains the QR code for sharing, while **Rectangle 3** holds the sharing link. **Rectangle 4** features buttons for copying and opening the link. **Rectangle 5** offers the option to create a poster for the survey, and **Rectangle 6** enables quick sharing through WeChat. Finally, **Rectangle 7** provides a way to conduct online face-to-face conversations with each participant.



Evaluate colorized images by various methods, right of the blue dotted line

1. For the colorized face image obtained by method A to method F, respectively. Evaluating the geometry preservation in the colorized images with respect to the **leftmost sketch image** in consideration of how well the colorized images maintain the geometric details and adapt the color into the sketch. **(5 points represent the best)**

Topic\Options	1	2	3	4	5
a	78 (78%)	19 (19%)	2(2%)	1(1%)	0(0%)
b	10(10%)	76 (76%)	13 (13%)	1(1%)	0(0%)
c	10(10%)	65(65%)	23 (23%)	2(2%)	0(0%)
d	2(2%)	1(1%)	20(20%)	70(70%)	7(7%)
e	1(1%)	3(3%)	6(6%)	76 (76%)	14 (14%)
f	1(1%)	2(2%)	9(9%)	16(16%)	72 (72%)
g	0(0%)	3(3%)	19 (19%)	54 (54%)	24 (24%)

Figure 4.9: **Rectangle 1** displays various essential information regarding the survey responses. From left to right, the options are: Statistics and Analysis, Download Survey, Source Verification, Completion Rate Analysis, and an Overview of all the responses. **Rectangle 2** offers the option to examine the data in more detail. **Rectangle 3** presents the distribution of votes across all scales for each question, as well as the percentage breakdown for each method. **Rectangle 4** illustrates different ways to visualize the data, including options such as tables, histograms, bar graphs, line graphs, and radar charts.

for each method across all the questions. Subsequently, we compiled the votes from these 30 questions and generated a comprehensive bar chart, which is illustrated in Figure 4.10. This chart provides a clear visual overview of the overall performance of each method.

4.6.3 User study analysis

Finally, we collected 100 feedback submissions online, leading to a total of 100 x 30 = 3,000 evaluation results for each method. The final statistical findings, as shown in Figure 4.10, clearly show that our method outperforms all competitors, receiving the most 'very satisfied' and 'satisfied' ratings for both geometry and style preservation.

However, we still observe a significant difference between the votes for geometry and color preservation. In geometry preservation, a few votes gave very low ratings to our method and others. This might be because the randomly chosen examples included some poor colorization results for less common facial attributes, such as hats or sunglasses. We will present these failure cases in the following chapter.

Moreover, we observe that RevGAN [82] mostly struggled to colorize the face sketch effectively, but it was able to keep a clear sketch face with minimal color variation. We think that some users may have appreciated that the sketch geometry does not change much from the input sketch, causing a few of them to give this method relatively high scores. Regarding color preservation, we notice that almost no one gave high scores to RevGAN [82].

We find that for both types of questions, participants displayed a clear preference for CNN-based methods [30, 50, 51] over existing normalizing flow-based methods [24, 59, 82] in the context of sketch colorization. This observation is also consistent with the results in our qualitative and quantitative comparisons. Furthermore, our method has received more positive feedback from participants, which highlights its superiority over other flow-based models and proves that the flow-based model is competent for the task of face sketch colorization.

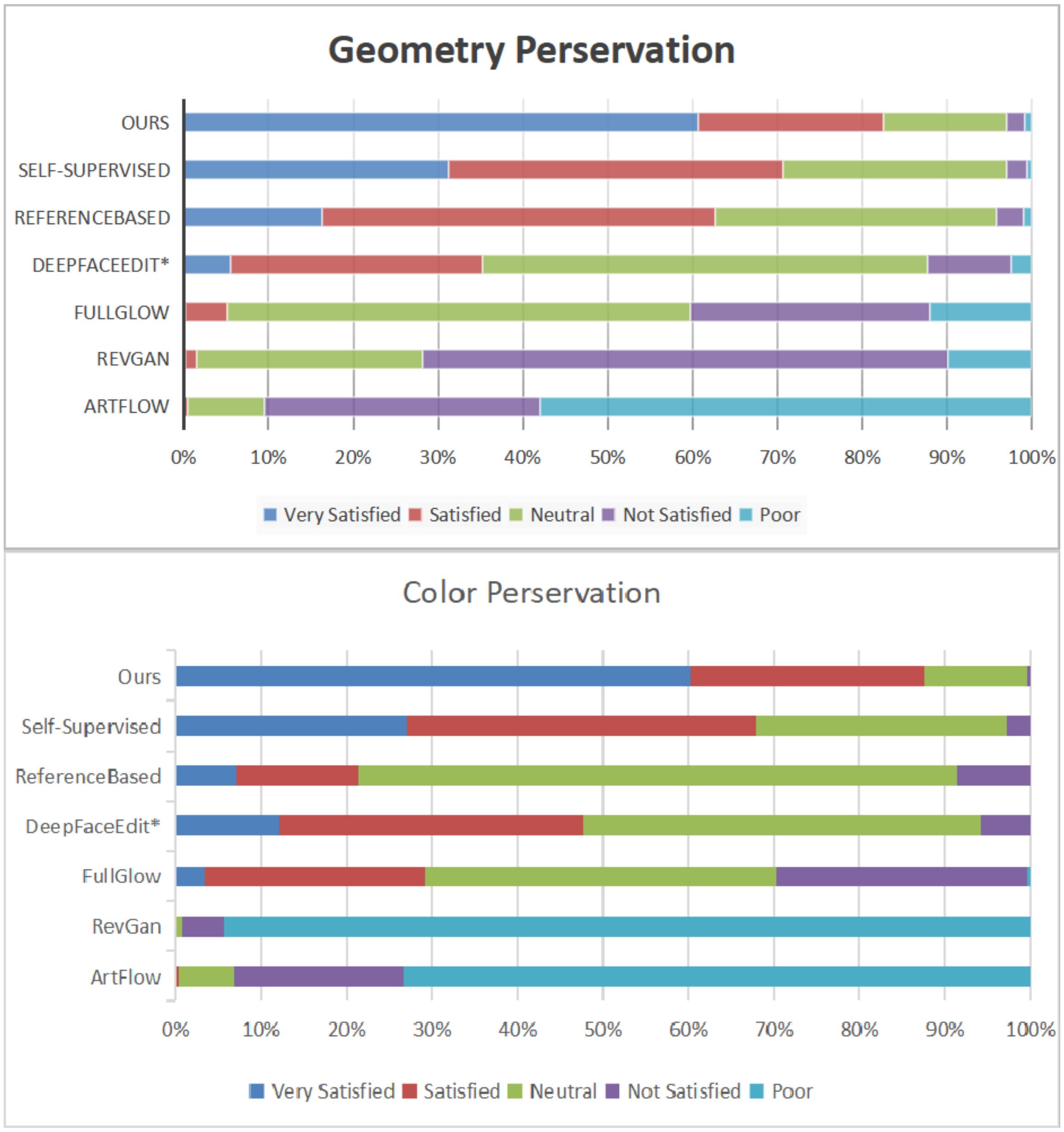


Figure 4.10: A summary of user study. In each table, the proportion of each rating has shown in five different colors such that 'Very Satisfied' and 'Poor' stand for the best and worst rating, respectively.

4.7 Summary

In this chapter, we present an overview of the methods we compared, including both CNN-based and flow-based models, as well as the four different quantitative evaluation metrics employed for sketch colorization tasks. Next, we present qualitative and quantitative evaluations, illustrating the satisfied performance of our flow-based method in accurately adding colors to face sketches while strictly preserving sketch geometry when compared to other state-of-the-art techniques. We also conduct ablation studies to validate the effectiveness of our proposed Feature Aggregation Module and demonstrate the memory-efficient design of our model architecture in compared to other flow-based models. Lastly, we present the survey responses from a user study, aiming to address aspects not covered by the previously mentioned metrics and allow individuals to offer more intuitive and perceptual assessments for a well-rounded understanding of our method's performance.

Chapter 5

Conclusions and future work

5.1 Thesis summary

Face sketch colorization involves accurately adding brightness and color information to a sketch while maintaining its geometry. This task can be seen as a combination of image colorization and image translation techniques. Numerous deep learning-based methods have been developed to tackle these two tasks separately.

In the case of image colorization, deep learning methods typically use Encoder-Decoder architecture to predict color information based on the available gray-scale information in the image. However, this approach may not be as effective when applied to a coarse sketch, which contains comparatively less information. On the other hand, GAN-based image translation methods often result in changes and imagination on the geometry in the final generated images, which affects the fidelity of the original sketch structure. Additionally, existing normalizing flow-based methods face challenges in establishing a effective correlation between the sketch and color information, leading to suboptimal results when translating between unpaired sketches and images.

To address these challenges, this thesis proposes an Encoder-Decoder architecture by using two normalizing flows composed of a stack of coupling blocks. First, we integrate a Feature Aggregation Module into the coupling blocks within a high-dimensional latent space, which helps improve the adaptation between color and sketch information. Then, we take advantage of learnable downsampling layers to reduce the model parameters related to shallow information. This method not only saves overall computational resources but also enhances the model's efficiency compared to other normalizing flow-based models.

To evaluate our approach, we use a comprehensive experimental framework that includes qualitative and quantitative assessments, along with a user study. The results indicate that our method not only successfully and appropriately adds colors to face sketches but also preserves the integrity of the original sketch structure. Our method yields visually satisfying colorized face sketches that receives many positive responses and achieves outstanding performance across various evaluation metrics, demonstrating the effectiveness and potential of our approach in the field of sketch colorization.

5.2 Limitation

Although our method has shown promising results and considerable progress in the field of facial sketch colorization, it is important to acknowledge some challenges that we faced during the experimental phase. In this section, we explore three main limitations of our approach:

The first limitation is related to our incapability to efficiently manage facial sketches with rare or uncommon attributes. For example, as shown in the first row of Figure 5.1, the sketches may include elements such as hats, which are not an inherent part of the face. When given a reference image without these objects, it becomes challenging to accurately predict the appropriate colors for such facial sketches. Likewise, when the reference image closely resembles the sketch but contains some rare attributes not present in the sketch, our method may struggle in dealing with these extreme cases. As illustrated in the second row of Figure 5.1, the flower in the girl's hair in the reference image leads to incorrect colorization of the sketch, where no decorations are present on her hair.

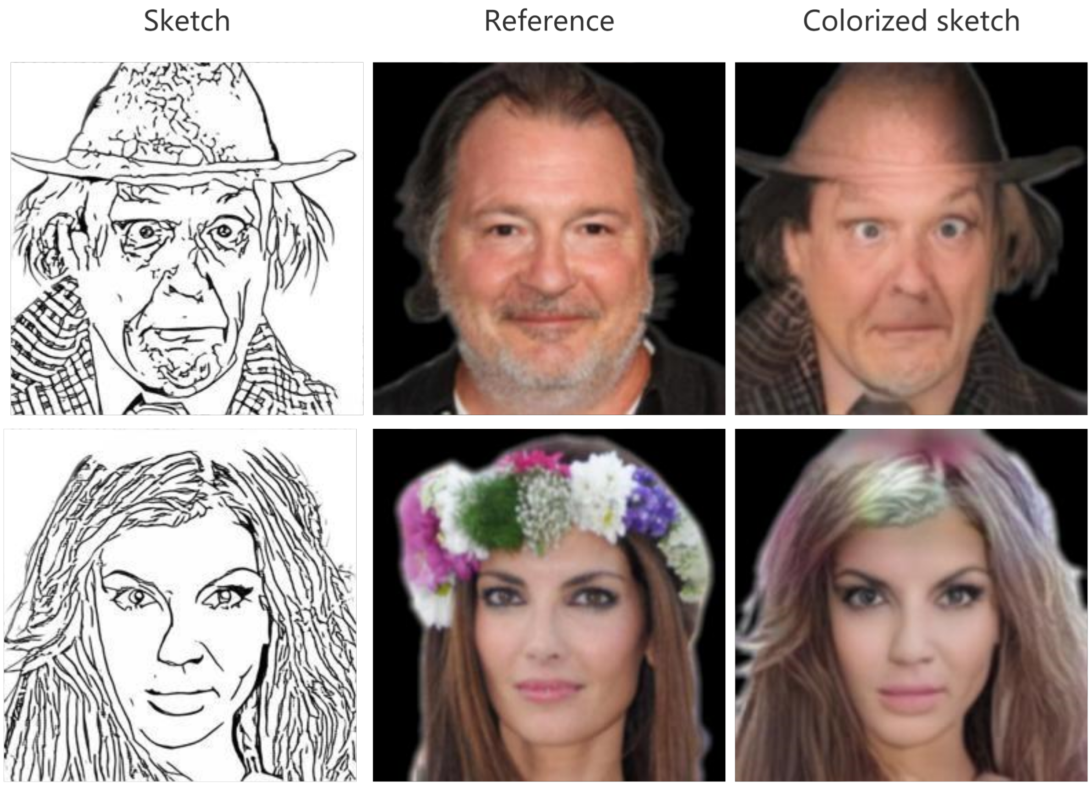


Figure 5.1: Examples of failure cases by our method.

The second limitation is related to the level of realism in our colorized images. When comparing image translation results by a pretrained StyleGAN [21], which transforms a coarse face sketch into a realistic face image, as displayed in Figure 1.6, the faces appear more realistic than those colorized using our methods shown in Figure 4.1b. One possible explanation is that the StyleGAN generator has been well-pretrained with lots of face or real images. As the pretraining can significantly improve the image generation quality by using some particular fine-tuning method, for complex objects and general scenes as suggested by Wang, et, al. [88]. Another factor is that pretraining is performed at a very high image resolution, such as 1024 x 1024 pixels. This additional detail captured by the model helps to improve its understanding of real faces, ultimately leading to increased realism in the final generated images.

The third limitation is our struggle to find suitable strategies for using normalizing flows to process high-resolution images, such as 1024 x 1024 pixels, within the constraints of limited GPU memory and training time. This challenge is due to the lossless encoding and decoding properties of normalizing flow, During the downsampling steps, when each time the spatial size of an image or latent features is halved, the channel information must be increased fourfold in response. For instance, with an image with a resolution of 3 x 1024 x 1024, the initial spatial size is 1024, and the channel size is 3. After three downsampling steps, the latent feature will have a spatial size of 128 and a channel size of 192. The processing of these features demands significant computational resources, which poses significant difficulty when conducting experiments with limited resources.

In the next section, we discuss potential future works and share some insights from our research journey over the past few years.

5.3 Future work

Recently, the remarkable success of GPT-4 [87] has attracted considerable attention due to its outstanding capabilities in generating human-like text and in understanding subtle differences in natural languages. Given the versatility of GPT-4 [87] and its capacity to handle a wide variety of tasks, we are interested by the underlying reasons for its success, as these insights could inspire and guide our future work in the field.

Pretraining, recognized as a key factor in GPT-4's [87] success, allows the model to achieve a deeper understanding of words across various language sequences. For the future work, we can apply a similar strategy for image processing tasks in our research. As discussed by Wang et al. [88], pretraining has the potential to help models develop a better understanding of complex objects or scenes within images. In the context of sketch colorization, it might be possible to implement a comprehensive pretraining phase for normalizing flow using a large dataset of face images and employing unsupervised learning techniques. As a result, when fine-tuning the pretrained model, it could generate more realistic faces, thus improving the quality of colorized sketches produced by the normalizing flows.

To prepare for the pretraining process, a larger and more diverse sketch dataset is crucial. As a result, another critical task involves gathering sketches generated from various methods and combining them to form a comprehensive dataset, as illustrated in Figure 5.2. The benefits of such an approach include enhancing the robustness of our model and improving its ability to handle a wider range of sketch styles. A similar strategy can also be applied to pretraining on real faces, such that it can better interpret any necessary details appears present in the real faces.

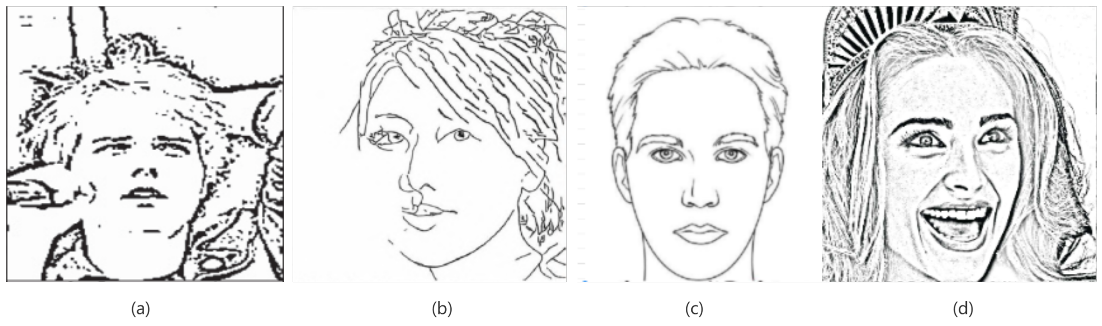


Figure 5.2: Examples of sketches generated using different methods. (a) is a sketch created by the approach proposed by Lee et al. [30], (b) is a sketch produced by Liu et al. [51], (c) is a sketch collected from online resources [51], and (d) is a sketch generated using the Photoshop sketch filter.

Subsequently, we also plan to add more user controls during the fine-tuning process, such as enabling the model to understand user-provided languages, color dots, and scribbles that further guide the colorization. This approach can improve the results when good reference images are unavailable, as these additional pieces of information can serve as a supplement or further refinement. This can be achieved by applying task-specific heads that project these different types of input information into the latent space that our pretrained model can understand. By integrating these user-centric features, it becomes possible for us to develop a more versatile model capable of meeting people's demands for a wider range of sketch colorizations.

Finally, in our future work, we also plan to generate high-resolution face images by integrating an additional model that undergoes a separate stage of training. This model will take inputs from our initial models, which have been effectively trained with lower-resolution images, and generate higher-resolution outputs. This approach is inspired by similar strategies employed by Wang et al. [88], who use a diffusion model for super-resolution tasks. Implementing such a model in our work could prove beneficial, as it allows us to avoid the significant computational cost typically associated with normalizing flow techniques when handling high-resolution images.

In addition to sketch colorization, we are keen on exploring the potential of flow-based models for a wider range of image processing tasks, such as the increasingly popular AI-driven face swapping techniques. Moreover, we also intend to investigate the applicability of flow-based models in tasks related to natural language processing. By delving into these additional applications, we aim to demonstrate the potential of flow-based models and contribute to expanding the horizons of artificial intelligence applications in our daily lives.

5.4 Journey of thought

During our research journey, we have gathered some insights and reflections that can serve as valuable advice for future researchers entering the field. If I were to start my thesis project again, I would first spend more time in determining the main contents of my thesis, which would help me understand which topics are easier to address and which ones are more challenging. This process is crucial because some topics may be very popular, with numerous high-quality papers available for reference, while others might be relatively unpopular, leading to a scarcity of related literature for guidance.

For instance, normalizing flow-based models have received less attention compared to other generative models, and the number of studies specifically focusing on image generation using flow-based models is even scarcer. A similar challenge exists in our chosen topic of face sketch colorization, where only a few works are directly related to the subject. Under such circumstances, we encountered numerous difficulties during our experiments, and it was often very struggling for me to solve those problems without any helpful materials available.

My advice to future students would be to initially focus on more accessible topics as a foundation. After gaining some experience, they can then explore more challenging subjects to further expand their knowledge and skills.

Another mistake I made during the period of my experiments was that I began building my models first and only later searched for other researchers' results to compare with. I overly optimistic assumed that the papers I had previously reviewed would provide comprehensive codes to generate their results. In reality, some papers did not release their training codes, or their codes were outdated and nearly impossible to execute. As a result, when we finished collecting our results, we had to abandon some methods that we initially intended to compare with. This forced us to spend a considerable amount of time searching for other related methods that had accessible codes for generating comparable results. In future research endeavors, it would be advisable to first confirm the availability of such resources before working on the model-building process.

Finally, it is worth noting some personal habit-related challenges that I faced during the research process. One such issue was my impatience while running code. I often came up with new ideas or identified new problems after starting the code execution, prompting me to interrupt the ongoing progress. This behavior resulted in a significant waste of time, as I often discovered issues with my code at the end of the week that could have been addressed earlier. Reflecting on this experience, I would recommend future researchers to practice patience and carefully plan their experiments first and to avoid such inefficiencies and maximize the effectiveness of their research efforts.

Bibliography

[1] P. Chen. “The dahuting tomb.” (2004), [Online]. Available: http://news.sina.com.cn/c/2004-08-25/08044129741.shtml.

[2] L. B. Isabella. “The yangtze valley and beyond.” (1898), [Online]. Available: http://collection.sina.com.cn/yxys/20150423/1245185676.shtml.

[3] C. Tian, Y. Xu, Z. Li, W. Zuo, L. Fei, and H. Liu, “Attention-guided cnn for image denoising,” *Neural Networks*, vol. 124, pp. 117–129, 2020.

[4] S. Anwar and N. Barnes, “Real image denoising with feature attention,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3155–3164.

[5] C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi, “Image super-resolution via iterative refinement,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[6] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, “Learning texture transformer network for image super-resolution,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 5791–5800.

[7] J. Li, N. Wang, L. Zhang, B. Du, and D. Tao, “Recurrent feature reasoning for image inpainting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7760–7768.

[8] X. Guo, H. Yang, and D. Huang, “Image inpainting via conditional texture and structure dual generation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 134–14 143.

[9] DGSpitzer. “Eking (beijing) in 100 years ago, ancient china (around1910-1920).” (5-12-20), [Online]. Available: https://www.goldthread2.com/culture/ai-old-beijing-video-color/article/3084049.

[10] J. Antic. “Jantic/deoldify: A deep learning based project for colorizing and restoring old images (and video!).” (Oct. 2019), [Online]. Available: https://github.com/jantic/DeOldify.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[12] A. Salmona, L. Bouza, and J. Delon, "Deoldify: A review and implementation of an automatic colorization method," *Image Processing On Line*, vol. 12, pp. 347–368, 2022.

[13] V. Manjunatha, M. Iyyer, J. Boyd-Graber, and L. Davis, "Learning to color from language," *arXiv preprint arXiv:1804.06026*, 2018.

[14] H. Bahng *et al.*, "Coloring with words: Guiding image colorization through text-based palette generation," in *Proceedings of the european conference on computer vision (eccv)*, 2018, pp. 431–447.

[15] R. Zhang *et al.*, "Real-time user-guided image colorization with learned deep priors," *arXiv preprint arXiv:1705.02999*, 2017.

[16] Y. Xiao, P. Zhou, Y. Zheng, and C.-S. Leung, "Interactive deep colorization using simultaneous global and local inputs," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2019, pp. 1887–1891.

[17] R. K. Gupta, A. Y.-S. Chia, D. Rajan, E. S. Ng, and H. Zhiyong, "Image colorization using similar images," in *Proceedings of the 20th ACM international conference on Multimedia*, 2012, pp. 369–378.

[18] J. Liao, Y. Yao, L. Yuan, G. Hua, and S. B. Kang, "Visual attribute transfer through deep image analogy," *arXiv preprint arXiv:1705.01088*, 2017.

[19] O. Frigo, N. Sabater, J. Delon, and P. Hellier, "Split and match: Example-based adaptive patch sampling for unsupervised style transfer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 553–561.

[20] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.

[21] E. Richardson *et al.*, "Encoding in style: A stylegan encoder for image-to-image translation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 2287–2296.

[22] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*, PMLR, 2015, pp. 1530–1538.

[23] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," *Advances in neural information processing systems*, vol. 31, 2018.

[24] J. An, S. Huang, Y. Song, D. Dou, W. Liu, and J. Luo, "Artflow: Unbiased image style transfer via reversible neural flows," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 862–871.

[25] I. Goodfellow *et al.*, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[26] S. Yang, Z. Wang, J. Liu, and Z. Guo, "Controllable sketch-to-image translation for robust face synthesis," *IEEE Transactions on Image Processing*, vol. 30, pp. 8797–8810, 2021.

[27] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2337–2346.

[28] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1501–1510.

[29] S. Liu *et al.*, "Adaattn: Revisit attention mechanism in arbitrary neural style transfer," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6649–6658.

[30] J. Lee, E. Kim, Y. Lee, D. Kim, J. Chang, and J. Choo, "Reference-based sketch image colorization using augmented-self reference and dense semantic correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5801–5810.

[31] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 415–423.

[32] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, Springer, 2016, pp. 649–666.

[33] F. M. Carlucci, P. Russo, and B. Caputo, "2 Co: Deep depth colorization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2386–2393, 2018.

[34] StabilityAI, *GitHub - Stability-AI/stablediffusion: High-Resolution Image Synthesis with Latent Diffusion Models — github.com*, https://github.com/Stability-AI/stablediffusion, [Accessed 01-May-2023].

[35] OpenAI, *DALL·E 2 — openai.com*, https://openai.com/product/dall-e-2, [Accessed 01-May-2023].

[36] Midjourney, *Midjourney — midjourney.com*, https://www.midjourney.com/home/?callbackUrl=%2Fapp%2F, [Accessed 01-May-2023].

[37] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, "Deep exemplar-based colorization," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–16, 2018.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[39] Z. Xu, T. Wang, F. Fang, Y. Sheng, and G. Zhang, "Stylization-based architecture for fast deep exemplar colorization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9363–9372.

[40] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[41] A. Alotaibi, "Deep generative adversarial networks for image-to-image translation: A review," *Symmetry*, vol. 12, no. 10, p. 1705, 2020.

[42] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

[43] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," *Advances in neural information processing systems*, vol. 30, 2017.

[44] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 35–51.

[45] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," *arXiv preprint arXiv:1610.07629*, 2016.

[46] H. Liu *et al.*, "Deflocnet: Deep image editing via flexible low-level controls," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 765–10 774.

[47] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.

[48] Y. Jing *et al.*, "Dynamic instance normalization for arbitrary style transfer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 4369–4376.

[49] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4401–4410.

[50] S.-Y. Chen *et al.*, "Deepfaceediting: Deep face generation and editing with disentangled geometry and appearance control," *arXiv preprint arXiv:2105.08935*, 2021.

[51] B. Liu, Y. Zhu, K. Song, and A. Elgammal, "Self-supervised sketch-to-image synthesis," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, 2021, pp. 2073–2081.

[52] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[53] P. Zhang, B. Zhang, D. Chen, L. Yuan, and F. Wen, "Cross-domain correspondence learning for exemplar-based image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5143–5153.

[54] Y. Wu, X. Wang, Y. Li, H. Zhang, X. Zhao, and Y. Shan, "Towards vivid and diverse image colorization with generative color prior," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 377–14 386.

[55] A. Lugmayr, M. Danelljan, L. Van Gool, and R. Timofte, "Srflow: Learning the super-resolution space with normalizing flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, Springer, 2020, pp. 715–732.

[56] J. Jing, X. Deng, M. Xu, J. Wang, and Z. Guan, "Hinet: Deep image hiding by invertible network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4733–4742.

[57] J.-J. Huang and P. L. Dragotti, "Winnet: Wavelet-inspired invertible network for image denoising," *IEEE Transactions on Image Processing*, vol. 31, pp. 4377–4392, 2022.

[58] A. Abdelhamed, M. A. Brubaker, and M. S. Brown, "Noise flow: Noise modeling with conditional normalizing flows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3165–3173.

[59] M. Sorkhei, G. E. Henter, and H. Kjellström, "Full-glow: Fully conditional glow for more realistic image generation," in *Pattern Recognition: 43rd DAGM German Conference, DAGM GCPR 2021, Bonn, Germany, September 28–October 1, 2021, Proceedings*, Springer, 2022, pp. 697–711.

[60] Y. Jo, S. Yang, and S. J. Kim, "Srflow-da: Super-resolution using normalizing flow with deep convolutional block," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 364–372.

[61] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pmlr, 2015, pp. 448–456.

[62] Y. Liu *et al.*, "Invertible denoising network: A light solution for real noise removal," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 365–13 374.

[63] L. Ardizzone, C. Lüth, J. Kruse, C. Rother, and U. Köthe, "Guided image generation with conditional invertible neural networks," *arXiv preprint arXiv:1907.02392*, 2019.

[64] A. Graps, "An introduction to wavelets," *IEEE computational science and engineering*, vol. 2, no. 2, pp. 50–61, 1995.

[65] C. Etmann, R. Ke, and C.-B. Schönlieb, "Iunets: Learnable invertible up-and downsampling for large-scale inverse problems," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, IEEE, 2020, pp. 1–6.

[66] A. Haar, "Zur theorie der orthogonalen funktionensysteme," *Mathematische Annalen*, vol. 71, no. 1, pp. 38–53, 1911.

[67] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5549–5558.

[68] S.-Y. Chen *et al.*, "Deepfaceediting: Deep face generation and editing with disentangled geometry and appearance control," *arXiv preprint arXiv:2105.08935*, 2021.

[69] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: Fully convolutional networks for rough sketch cleanup," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–11, 2016.

[70] H.-Y. Lee *et al.*, "Drit++: Diverse image-to-image translation via disentangled representations," *International Journal of Computer Vision*, vol. 128, no. 10, pp. 2402–2417, 2020.

[71] W. Xian *et al.*, "Texturegan: Controlling deep image synthesis with texture patches," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8456–8465.

[72] J.-W. Su, H.-K. Chu, and J.-B. Huang, "Instance-aware image colorization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7968–7977.

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

[74] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[75] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *International conference on medical image computing and computer-assisted intervention*, Springer, 2016, pp. 424–432.

[76] P. Y. Simard, D. Steinkraus, J. C. Platt, *et al.*, "Best practices for convolutional neural networks applied to visual document analysis.," in *Icdar*, Edinburgh, vol. 3, 2003.

[77] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998.

[78] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, Springer, 2016, pp. 694–711.

[79] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[80] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[81] K. Fukushima, “Visual feature extraction by a multilayered network of analog threshold elements,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 5, no. 4, pp. 322–333, 1969.

[82] T. F. van der Ouderaa and D. E. Worrall, “Reversible gans for memory-efficient image-to-image translation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4720–4728.

[83] Z. Liu, P. Luo, X. Wang, and X. Tang, “Large-scale celebfaces attributes (celeba) dataset,” *Retrieved August*, vol. 15, no. 2018, p. 11, 2018.

[84] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” *Advances in neural information processing systems*, vol. 30, 2017.

[85] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[86] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[87] OpenAI, *Gpt-4 technical report*, 2023. eprint: arXiv:2303.08774.

[88] T. Wang *et al.*, “Pretraining is all you need for image-to-image translation,” *arXiv preprint arXiv:2205.12952*, 2022.

Appendix A: Appendix

We provide an extra qualitative comparisons between our method and others in Figure A.1.

We present more colorization examples using our approach in Figure A.2, A.3, A.4, A.5 and A.6, while these Figures takes the same caption: Qualitative comparison of our method that each sketch image (in the first column) takes 8 colorful images (in the first row) as exemplars.

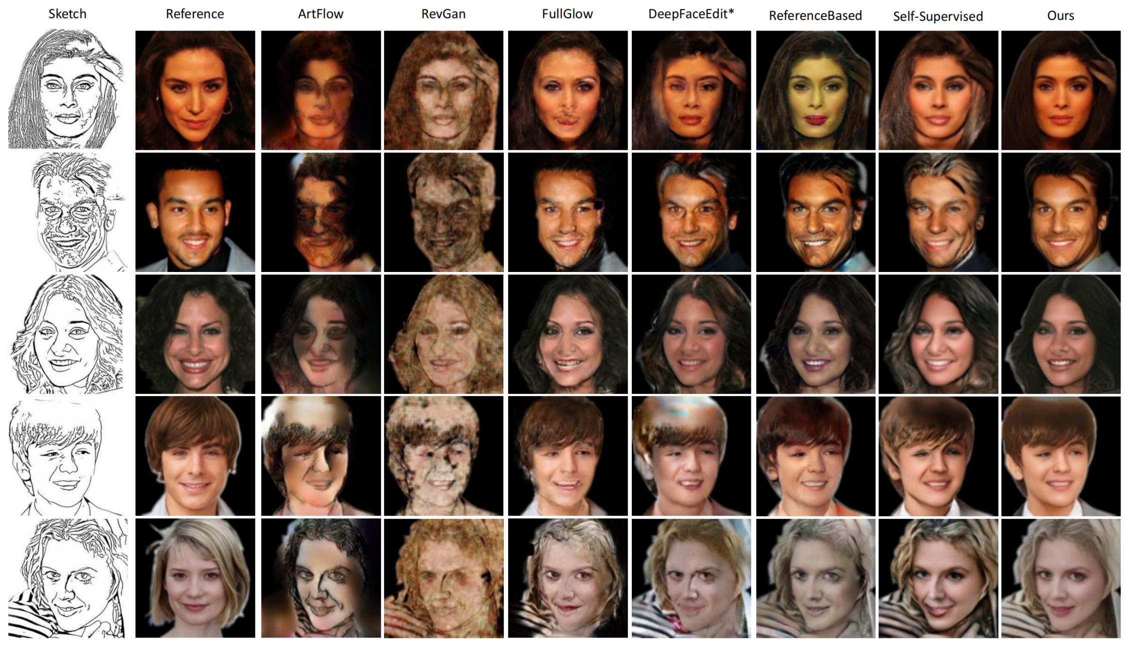


Figure A.1: Qualitative comparison against with other baseline methods, The first column is the input sketch image and the second column is the corresponding reference image. The last column is the colorized results using our method.

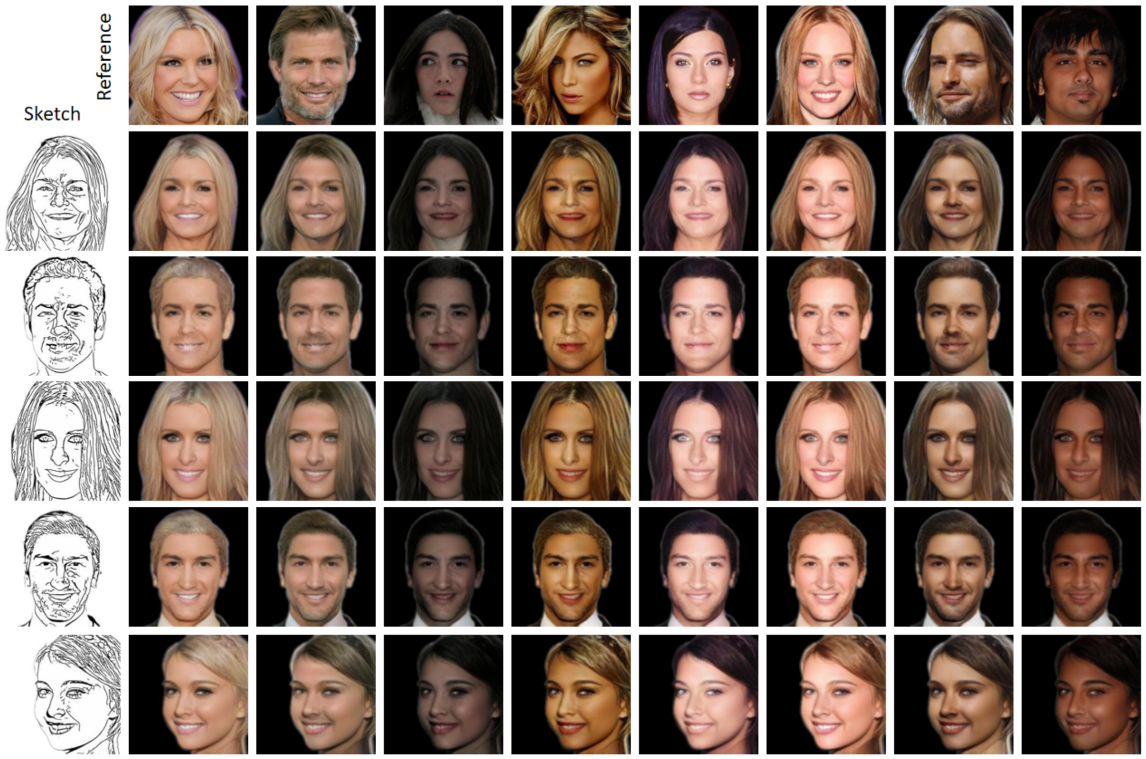


Figure A.2

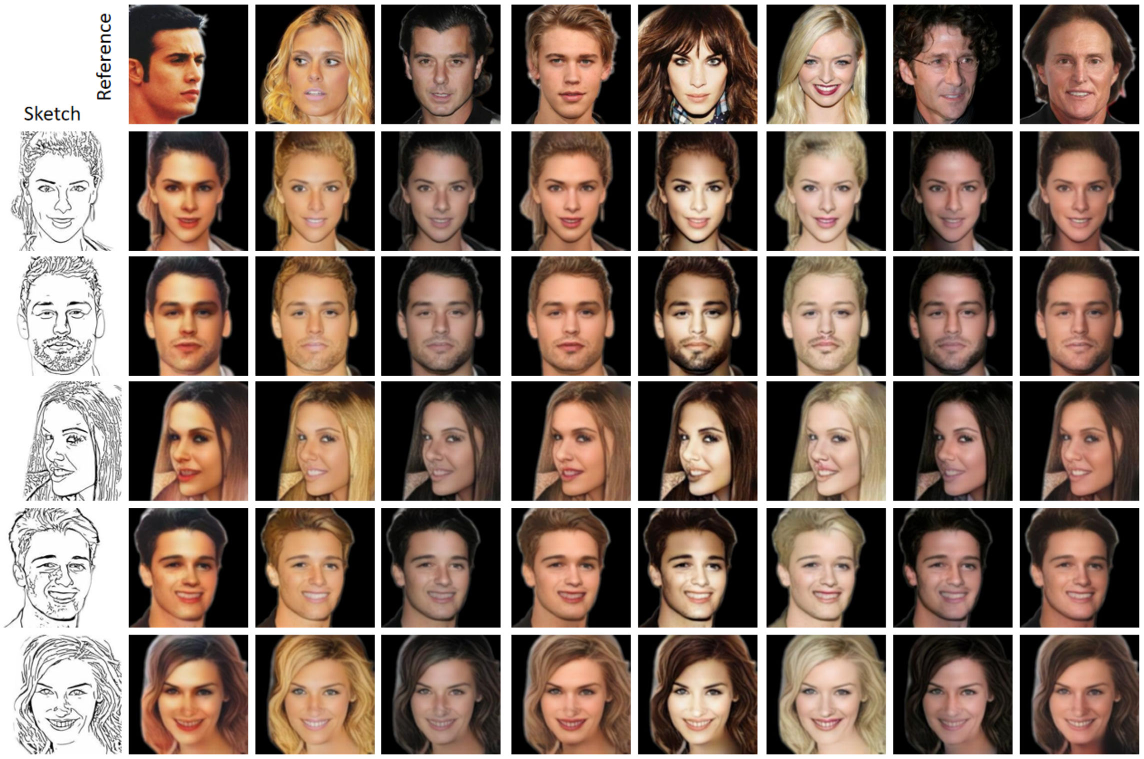


Figure A.3

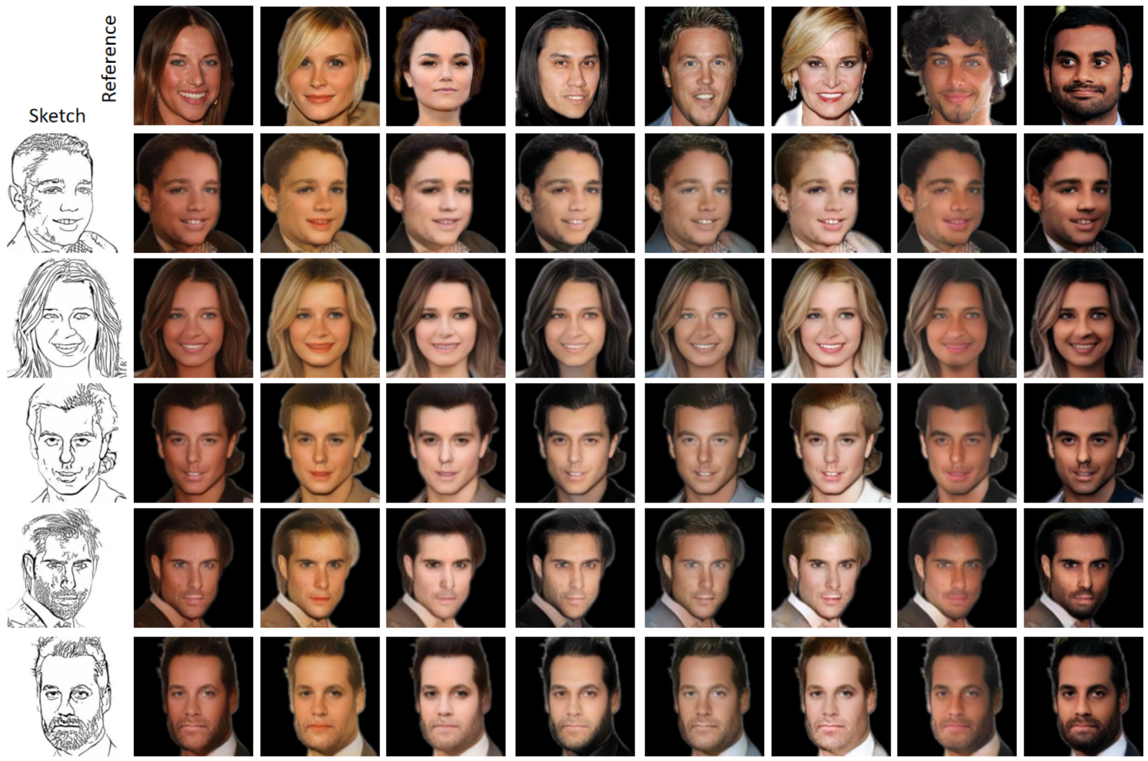


Figure A.4

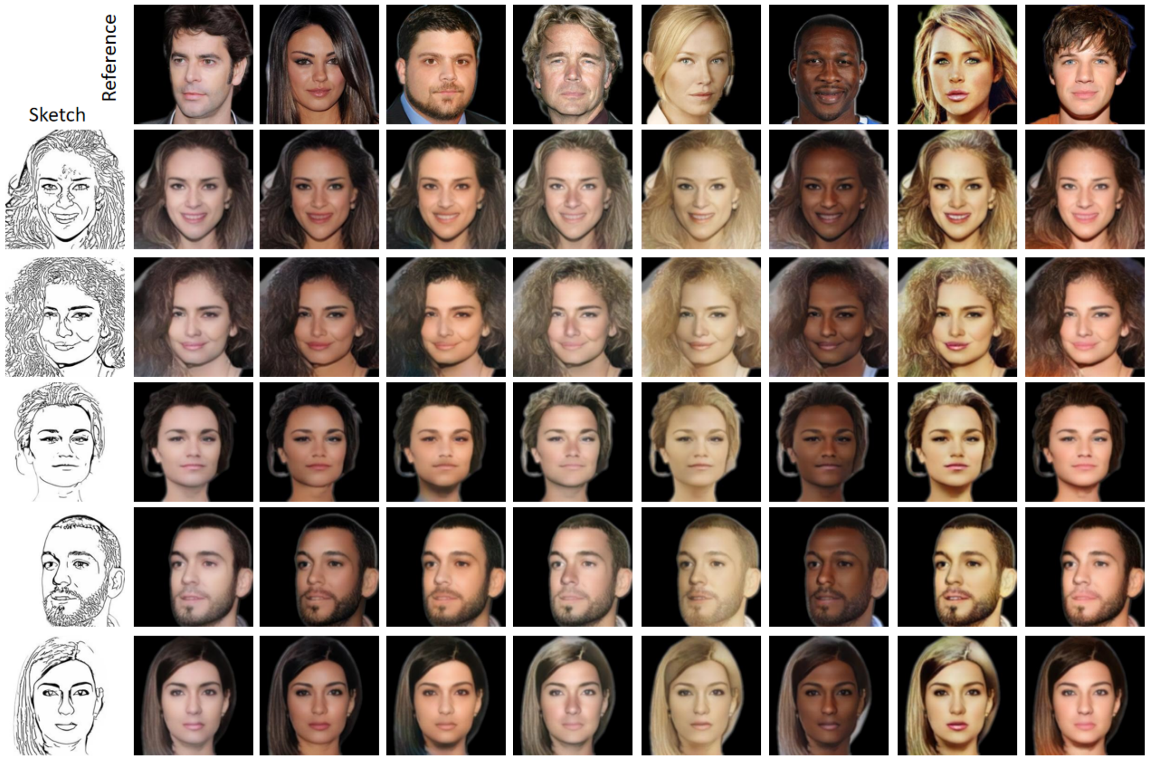


Figure A.5

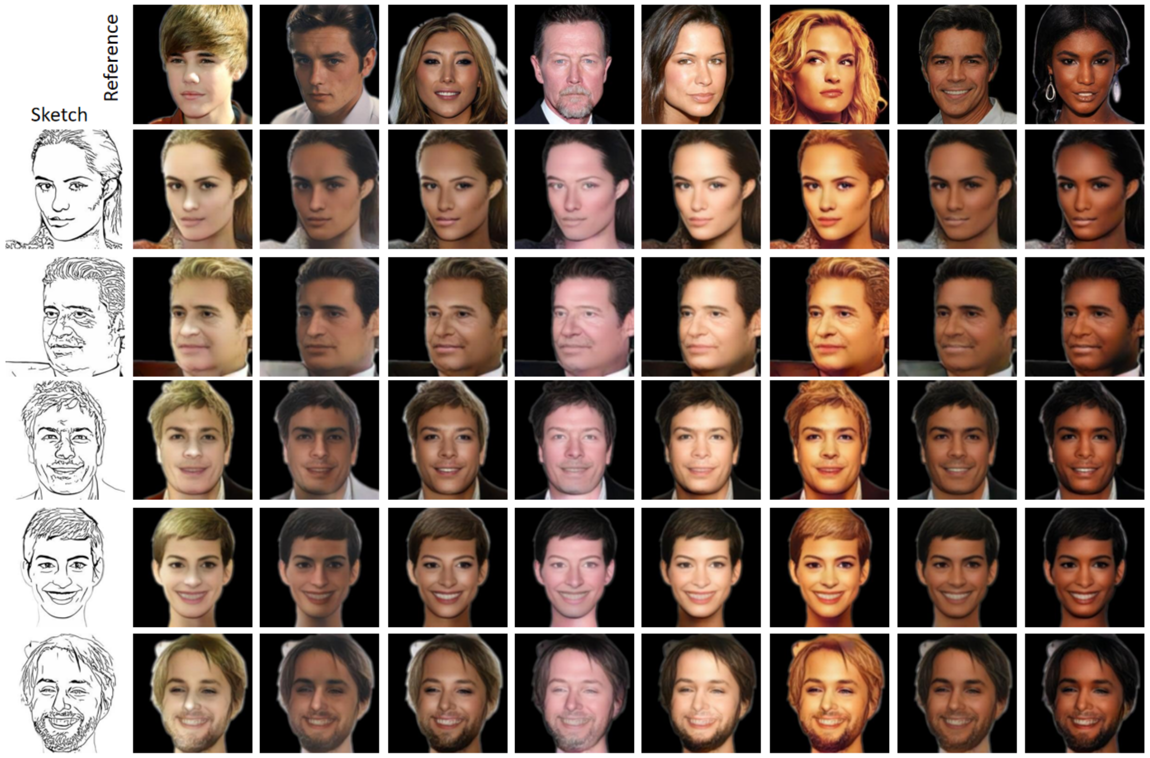


Figure A.6