National College of Ireland

Image Completion: Two-Staged Architecture

MSc Research Project
MSc. in Data Analytics

Hsin-Liang Liu
Student ID: x19116829

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Hsin-Liang Liu
Student ID:	x19116829
Programme:	MSc. in Data Analytics
Year:	2020
Module:	MSc Research Project
Supervisor:	Christian Horn
Submission Due Date:	17/08/2020
Project Title:	Image Completion: Two-Staged Architecture
Word Count:	XXX
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	16th August 2020

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Image Completion: Two-Staged Architecture

Hsin-Liang Liu
x19116829

Abstract

Image completion or image inpainting refers to a technique that is to restore the images or remove unwanted objects and fill it with something that might have been there before. There are many methods for image inpainting for example the patch-based image completion. It searches the data and finds the patch that match the hole the best and complete the image with that patch. Nevertheless, it, sometimes, does work well and produce the unsatisfactory results. The patch can fit well in the hole for certain contents of the pictures but it fail once the contents are changed. In order to improve the performance and output the content-aware pictures, this research proposes a method that aims to provide more information about the missing pixels based on the symmetry of the face before it propagate to the neural network. Most of the current deep learning network take the damaged images and try to construct the new images without any knowledge about the missing portion of the images. Taking advantage of the face structures, we can improve the current algorithm so it get semantic image completion results. Our experiments show that it could lead to 5% improvements in term of SSIM and PSNR that measure the similarity of the two pictures. This research implements a dual-pipeline network structure(Zheng et al.; 2019) and this methodology can also be applied to other deep learning methods in an effort to improve the performance.

1 Introduction

The image completion application can be traced back to be used for art restoration (G. and M.; 2014) and even until today it is still popular and active research area because the increasing in demand for digital removing unwanted objects. The techniques are developed to fill the hole or gap of the image to looks faultless. Not only the 2D images, but it can also be applied to 3D images. The challenge of image completion is to recover the damaged images while keeping the consistency of images using the content-aware algorithm. I have been working on the image completion project since a while ago and have tried to use it for recovering face images. This project is dedicated to open a new possibility of the new image inpainting on top of the current techniques. Conditional Variational Autoencoder (CVAE) has been widely used as generator to create images according to requirement and has achieved state-of-art results. However, it has limitation when it is used for image completion when parts of the images is missing and the CVAE has to learn to adopt the situation and output the complete images. This is why I propose a new algorithm to compromise this situation.

The first section gives gives an overviews of previous work that is related to the project especially the work that inspired me to come up novel idea for improving CVAE

in the report. In the following section, we show how to preprocess the data and what are the challenges of handling large-scale datasets. The data contain more than 23,000 3D arrays that represent 23,000 face images. If I load the such large-scale raw data all at once, it pose the threat to our system which is necessary for us to preprocess the data beforehand. Next, I will present you what I learn when I survey the VAE knowledge and how the system is designed. Our image completion structures consist of two stages that were devised for image completion but with more face information comparing to the previous approach. Face are highly structured and somewhat symmetric, it give us insights to utilize the symmetric characteristic to pre-construct the missing area of the image in the effort to yields a better performance. In the current deep learning approach, the computer can recognise a person by only half of the face image (Singh and Nandi; 2012). It takes advantage of the face symmetry and the computation cost is lower because it only requires half-face image instead of full-face images during training and testing. We referenced the dual-pipeline structures to build our deep learning models along with our face construction block and random mask block to generate the new images. Result are seen in the next section where we show the outcome and how our proposal's performance compare to others in term of PSNR and SSIM. Due to the development of deep learning methodology, the image completion has achieved significant improvement in the recent decades. However, it is still a long way to go because the difficulty of filling the hole that is semantic consistency with the rest of the images.

2 Related Work

Many deep learning techniques that apply to images and videos involving images preprocessing which aims for better outcome instead of working with raw data. For instance, we can apply image segmentation, filtering and color conversion to the images before we feed them to the neural network for image classification or directly using deep convolutional network for image denosing (Amara et al.; 2017) (Wang et al.; 2015). It plays an important role in normalizing and regulating data and removing background noise. We set the image size to be 128×128 with 3 channels and normalized the value to be between 0 and 1 before feeding data to the two-stage architecture. Our proposed methods try to generate the complement images of the damaged images in the first stage. Although it is from completely different concept, path-based methods are also trying to find the patch to fill in the missing region of the images. Considering the previous path-based methods fail in generating coherent images, an improved PatchMatch technique is proposed that construct mid-level structure as a guidance for image completion against various contents of images (Huang et al.; 2014). In addition to filling the hole, (He and Sun; 2014) find the relative position of the patch that could offer valuable information for image inpaining assuming that the relative position are statistically distributed. Giving a face image, human usually are able to recognise the pattern of the face for example the eyes are

above the nose. Similarly, we construct part of the missing pixels using the knowledge of the pattern so we can have more information for the neural network. We can imagine that before the deep learning methods come to place, this patch-based methods is widely used to solve the image inpainting intuitively. In order to increase the efficiency, we can use the low-rank approximation where we convert multiple patches to form a low-rank path-matrix for further processing to find the best matched patch (Guo et al.; 2018). Multiple stage of sequence of image processing was employed to yields better performance when restoring images. Furthermore, many researcher develop image completion by combining different technology including patch-match such as patch-based methods for finding k-nearest correspondent patches (Liu et al.; 2015).

Next, we will talk about face parsing which is tightly related to our image processing 2.1. The main components of this stage includes random mask block and patch construction block. In 2.2 we survey the current deep learning methods that are for image completion and in 2.3 the works related to dual-pipeline network in pluralistic image completion.

2.1 Face Symmetry and Face Parsing

Facial recognition has long been a popular topic driven by the market. It can be used to track criminal or to record people's identity. As we mentioned earlier the low-rank patchMatch techniques, it could apply to facial recognition if we treat the patch as units and find the correlation between them (Jiang et al.; 2017). Although no one has perfect symmetric face, we are under the impression that the human face has a symmetry in our experience. It is being used as a biometric trait mark and face recognition. Due to the face symmetric features, (Singh and Nandi; 2012) present a technique that even half-face image is enough for identifying a person using PCA. Applying their proposed algorithm to train with large data of full and half-face, the accuracy difference between by full-face and half-face is slim, but it could lead to smaller recognition resource and smaller time complexity with half-face data. There are many application developed by using the symmetric structures of faces. For example we can get the axis of the face by detecting the symmetry of the face.

Using the human parsing and pose estimation to handle human body part missing image achieves significant improving comparing with existing algorithm in term of PSNR, SSIM (Wu et al.; 2019). In the first stage, it apply human parsing to recognise the human body pattern and segment those part from the rest of the images. The parsing subnet and pose subnet are mean for creating the parsing map and pose heatmap which will further propagate the signals to the refinement subnet for human body part estimation. Inspired by the pose estimation and human parsing, I construct and implement a two-stage architecture for human face completion. Like (Wu et al.; 2019), our first stage is to recover part of the missing values by finding the patch on the other half face that match the hole the best in order to decrease the error when recovering the images. The facial features are composed of many distinguished components such as eyes and noses. In recent years, partial face recognition has drawn people's attention since not all the images can get the full-face directed in the camera, it is convenient for the deep learning recognising the face components and being able to reconstruct them (Elmahmudi and Ugail; 2019). It extract and match the facial features to the images in the dataset. Similarly, we can start by detecting the key points of the facial features and search the patch the match local textural features for partial face recognition (Weng et al.; 2016). (Cai et al.; 2019)

approach the problem differently through multi-task learning. The FCSR-GAN first train the face completion module with the face completion loss, then it trains face completion, super-resolution module and the discriminator with face completion loss, super-resolution loss and adversarity loss to have joint face completion. This partial face recognition using patch that match the missing give us insight to repair damaged images. We perform eyes detection to find the center line of the face then we can get the patch from the other half-face.

However, Completing human faces still remains a challenge due to the complexity and highly integrated structure of face. Other than patch matching, many developers also work on creating semantic consistent face image (Deng et al.; 2011) (Li et al.; 2017) using learning-based methods in the past decade. Beside a generator and discriminator, (Li et al.; 2017) proposed an algorithm with extra generator and discriminator to further validate the facial features images. It is called parsing network which is learned to recognize facial features such as nose, eyes and ears. Moreover, most of the learning-based methods such as (Zheng et al.; 2019) (Huang et al.; 2014) is effective to fill the hole of the images under certain condition, but with patch matching enhancement or guidance it could further impove the performance make the outcome more realistic and reliable.

2.2 Learning-based Approach

VAE has been used for image inpainting for years. For instance, the cosmoVAE where each encoder and decoder have six blocks was presented to uncover the missing region of Cosmic microwave background radiation (CMB) images which contains the information of universe's early creation (Yi et al.; 2020). Connected the decoder and encoder with the latent space, the U-net like architecture of VAE with each layers from encoder also feed-forward signals to the decoder was devised to reconstruct the missing observations. Or we can change how we sample the latent variables and make many channels that process the variation of latent variables between inference network and generation network to create the channel-recurrent network (Wenling Shang and Tian; 2018). These design allows more details of the images to be able to pass through the latent space via the channel connections. Although, it is still an on-going research area, these variation of VAE has reach certain level of promising results when it comes to image completion task in different condition. Similarly, our architecture has channels but they are meant for forwarding the signal from encoder to decoder to bypass the latent space. The challenge of image completion is to synthesis the missing part of the image with content-aware and semantic consistency. To overcome this problem, (Wei et al.; 2019) proposed a new model that combine the patchMatch and VAE. Once VAE output coarse images, it search patch from the face images dataset to fill the hole with the best match.

The CVAE learning algorithm is a special version of VAE. It allows us to bend the learning output to certain region by introducing an controlling variable. In the development of CVAE, (Sohn et al.; 2015) proposed a new generative model that maximize the log-likelihood of the result conditionally. It makes the subset of output conditional to the controlling variable. The proposed recurrent model from (Sun et al.; 2016) can generate multi-digit images according to sequence of numbers for example. The controlling observation characterize the latent variable over the sparse latent space so the user can manipulate what type of output they want to generates. In VAE, the latent variable z is strictly based on the input x and lack of information such as what category the input x is. At the same time, the output y is based on the latent variable so the input x has

no control over what type of output x' is. This situation can be changed by introduce a conditional variable c into the system. So the encoder is conditioned to x, c that is $q(z|x, c)$ and decoder is $p(x|z, c)$ and the users are able to control the distribution via the c. In our project, it has many interesting facts and leave us some area to explore such as how do we decide the controlling variable z is. Currently, we take damaged images x and complement image c as the inputs where damaged images is denoted as I_m and complement images is I_c. Through random generation of I_c, we want to maximize the performance after several rounds of training. In (Zhang et al.; 2020), it is used for saliency detection to find what is the most relevant in the images or it can be used for image colorization in which the grey image is the input x and colored image is the c (Deshpande et al.; 2017). With different loss function implementation, it can lead to different image colorizing effects. In our project, we also try different loss function to see which one has satisfied results.

Many learning-based image completion techniques that utilize VAE as generator will pair with discriminator to improve the performance (Zheng et al.; 2019). A global and local discriminator was proposed to distinguish between real images and the image after image inpainting to reach globally and locally consistent where the global discriminator works with the image as as whole and local discriminator looks more details in the completed region (Iizuka et al.; 2017). Specifically, (Li et al.; 2017) devised a structures that also adopt GAN and VAE to do the face image completion where local discriminator is used to evaluate missing region of the images while the global discriminator is for the whole-images.

2.3 Two-Staged Approach

Prior to commence our project, we learn that the traditional neural network lack of diversity when it comes to training an image because every image corresponds to only one training instance while the gap or hole could appear in different area and in different shape and size in the image. There are millions way of how the image damaged but we only update weights based on the information from the complete and full image per label. In another word, it is hard and challenges to cover all the learning cases for each damaged image. The training data or label to the corrupted or incomplete images is one to many relationship instead of one to one relationship. In order to tackle this problem, a parallel pipeline network was proposed to compromise the only one training instances per label (Zheng et al.; 2019). It uses the extra pipeline to process the complement of mask images information (I_c) during the training with the forward long short term from the encoder. It is shown effective and able to is create high quality of reconstructed images in comparison with the PatchMatching such as using random sample to search the match (Barnes et al.; 2009) (G. and M.; 2014) and other learning-based methods.

One of the issue about image completion come from that the previous learning-based models do not distinguish the missing pixels from the remaining pixels. In another word, it treats all the pixels include hole or damaged part of images as valid ones. Therefore, free-formed mask was introduce SN-patchGAN which is based on gate convolutional network for filling the hole of the image (Yu et al.; 2019) or apply partial convolution where we only works on the non-blank pixels leaving the hole invalid (Liu et al.; 2018). Another example is (Yu et al.; 2018) which uses the advantage of the traditional patch and texture filling approaches for various holes with different size and location. Similar to the free-formed mask generation, we build mask generation block that works on generating images

with hole on the fly during the training to diversify the output and train the models so it learns how complement images can contribute to the output. This step is crucial because the images with random hole signify the numerous possibility how the images could be damaged and we trains the models to adopt that situations.

In many modified neural network, progressive image completion with with residual block and dilation layers has been proven effective such as full-resolution residual network (FRRN) (Guo et al.; 2019). It also provide two parallel path which consists mainly of partial convolutional layers that trains both damaged images and mask images. In recent years, learnable attention architectures was presented such that the new architectures is adopt to unpredictable masks and it is trained to recover the image along with the mask (Xie et al.; 2019). The novelty of this learnable attention map is that it is not a one way but bidirectional map where the mask and inverse mask are propagated in opposite direction and each layers will get multiple inputs including the signal from the neural network that is trained with mask. The decoders will take signal from the learnable reverse attention map to get the mask information which is the dot product of the mask and the filters. The bidirectional attention map is developed from U-net in which the convolutional layers are without bias. Another form progressive image completion is combining convolution and recurrent neural network which imitate how human solve difficult problem. The Recurrent Feature Reasoning (RFR) (Li et al.; 2020) is designed to infer the missing region of the image and reconstruct it. The architectures can be roughly categorized as Area Identification, Feature Reasoning and the Adaptive Feature Merging. The main task of area identification is segregating the area that needed to be inferred from the remaining images with the partial convolutional layers. The core of feature reasoning is a modified autoencoder (generator) with a Knowledge Consistent Attention (KCA). The output of the feature reasoning will be feed to the area identification recurrently. The feature merging operator then further merge the valid intermediate features map to reconstruct the original images. In recent years, progressive neural network architecture and corresponding application has been widely studied. In order to tackle the complexity of restoring face images, a deep learning models is developed with the guidance of facial map guidance after the estimation of missing pixels (Yang et al.; 2019). The models contains two different pipeline for processing the facial landmarks and generating new face images that also adopt feed-forward passage from encoder to decoder. It use landmark prediction module to predict landmark before it feeds to the reconstructive network which is related to our two-stage architecture. Different pipeline in deep network for different purpose trigger many interesting research area. Take blind image inpainting for instance, we can have pipeline for predicting where mask is and feed-forward the information for synthesizing the missing part of the image.

3 Methodology

As we know the traditional AE which composed of Encoder and Decoder is sufficient enough to copy images, but it is not ideal when it comes to generating new images. It is because in the latent variable that are from AE are not continuous and it is very likely that the image we generate we generate does not make sense at all. For traditional AE, we define the loss function $L(x, x^{'})$ which is the reconstruction error between the original input x and the output $x^{'}$. The goal is to minimize the loss function so the distance between the input and output data is close as possible. The AE has the ability

to compress data because the data has to propagate through the bottleneck (latent space) that is smaller than the original size of the data and being decoded by the decoder. The AE is basically a CNN that learn to extract the important features from the images and make it compact enough to go through the latent space. It can be used to denoise images.

Variational AE is superior than traditional AE when it comes to generate new pictures because VAE sample from distribution(ex. Gaussian distribution) which is more likely to interpolate the data that is meaningful to the decoder. The encoder output means μ and standard deviations σ together($Z_{\backsim N(\mu,\sigma^2)}$) that result in stochastic behavior which enforce decoder to learn that the data in certain region is associated with certain class. Moreover, we add the KL-divergence loss function term so the probability distribution will be close to each other. The general KL-divergence, short for Kullback–Leibler divergence or related entropy, is represented as $D_{KL}(P||Q) = \Sigma p(x) \log \frac{p(x)}{q(x)}$. From there as you can see the KL-divergence will converge to zero when p(x) is close q(x). It used to measure how similar the two distribution is. Furthermore, we use **reparameterization** trick so it is able to do backpropagation under this structure(Vasilev et al.; 2019).

$$z = \mu + \sigma \odot \varepsilon$$

When I construct model and learn the VAE methodology, I reference a lot of citation from the book (Vasilev et al.; 2019), it use $q_\psi(z|x)$ and $p_\theta(x|z)$ to denote the encoder and decoder where x is the data and z is the latent variable and ψ and θ are the weight and bias, respectively.

- Encoder: Encode the information so the information can be map to the latent variable z ($q_\psi(z|x)$)
- Decoder: Decode from the latent variable or latent representation and try to reconstruct the original information($p_\theta(x|z)$)

For our project, the encoder and decoder are constructed of even smaller subencoder and subdecoder such as E1, D3. The loss function for the VAE is

$$L(\theta, \phi, x) = -D_{KL}(q_\psi(z|x)||p_\theta(x|z)) + \mathbb{E}_{q_\psi(z|x)}[\log p_\theta(x|z)] \tag{1}$$

The KL-divergence term can be viewed as the measurement of the difference between the $q_\psi(z|x)$ and the $p(z)$ distribution. It is for minimizing the difference and bring the two distribution together (Vasilev et al.; 2019). The second term is the construction loss because we want the $q_\psi(z|x)$ to be close enough to represent $p_\theta(x|z)$.

For the CVAE, it introduce another variable c to conditioned the output, it is usually append the input x and latent representation z. As I mentioned earlier, it has the ability for the data that has similar pattern to cluster each other so we are able to output the image from the certain group of class in the latent space. In this project, I use the same notation as (Zheng et al.; 2019) for simplicity. I_m represents as input for the corrected or damaged images; I_c is the complement images of I_m so the I_c can also be viewed as conditional variable. Unlike I_m where image is degraded and damaged, I_g is the original images and it is the target when we reconstruct I_m. Generally, we take both I_m and I_c as our input and our model will output $I_g^{'}$ which is used to fill the hole of the I_m. On top of that, we use λ_6 to weight the standard deviation of z_c, it is because the partial images of either I_m or I_c has different numbers of pixels, and the images(ex. I_m) that has more pixels deserves to have greater latent variance than the images(I_c) has fewer pixels.

$$
\begin{aligned}
z_{I_m} &\sim N(0, (1-\lambda_6)\sigma) \\
z_{I_c} &\sim N(0, \lambda_6\sigma)
\end{aligned}
\tag{2}
$$

Given by the lower bound of the log-likelihood of the instance for their CVAE model(Sohn et al.; 2015):

$$
\log p(I_c|I_m) \geq -KL(q_\psi(z_c|I_c, I_m)||p_\phi(z_c|I_m)) + \mathbb{E}_{q_\psi(z_c|I_c,I_m)}[\log p_\theta(I_c|z_c, I_m)] \tag{3}
$$

we can see that the overall loss has minimum value. We can assume that if the I_c and I_m are strongly related to each other, it bring the total loss down. According to the (Zheng et al.; 2019), z_c is more related to I_c instead of I_m so we can approximate $q_\psi(z_c|I_c, I_m) \approx q_\psi(z_c|I_c)$. Therefore we have

$$
\log p(I_c|I_m) \geq -\lambda_4 KL(q_\psi||p_{phi}) + (1-\lambda_4)\mathbb{E}_{q_\psi}[\log p_\theta(I_c|z_c, I_m)]) \tag{4}
$$

where λ_4 is a value between 0 and 1 which I used to adjust the power of KL-divergence and construction loss. Specifically, we have three term that form our loss function.

$$
L = \lambda_4 L_{KL(z_{I_c}||z_{I_m})} + L_{KL(p||q)} + L_{\mathbb{E}} \tag{5}
$$

The first term is for narrowing the difference between the latent representation of I_c and I_m because we assume that the similar between z_{I_c} and z_{I_m} can lead to smaller lower bound of loss function according to the equ. 3. The second term is to bring two distribution close together so the latent variable will lies continuously in the latent space. The last one is the reconstruction loss where we want our generated images are alike to the original images and are closely related to each other. After we construct the $I_g^{'}$, we use the patch of the related position of it to fill the hole. If $I_g^{'} \approx I_g$, then the hole should be filled seamlessly and have semantic consistency.

4 Data Processing

We use the cropped face images for our data that we found via the citation of papers or directly from the internet. It is from open source or celebrity face images. Two main source of face images are UTKFace and FEI Face Database and for non-commercial usages. UTKFace has more variation both in quality of the images and the facial expression, but the FEI Face Database is more concise and all the pictures are from the same lab in São Paulo. Therefore, we mainly use FEI Face for evaluation when it comes to testing and prediction.

UTKFace UTKFace dataset contains around 23k cropped and aligned face images in a zip (Zhang et al.; 2017)[1]. The images were taken from wide variety of people of different ages (0-104), different ethnic group, man and women. The file name for each images represents age, gender, race and time in a format of age_gender_race_time.jpg (Zhang et al.; 2017) so we can group each images according to the file name(labels). The images are usually frontal face images but some of them are shot in different angle and with different facial expression. Because the data is extract from the internet, they have different resolutions and image size even for the same person.

[1]https://susanqq.github.io/UTKFace/

Figure 1: Example of UTKFace Data

FEI Face FEI Face database has 400 frontal face image that were taken from 200 person of different gender and different expression at Artificial Intelligence Laboratory of Universitario da FEI, São Paulo between June 2005 and March 2006 (Thomaz; 2006)[2]. The images are colored and are of 360×260 pixels that were cropped from the original 640×480 images with natural white background (Thomaz; 2006). Each person in that images has neutral and smiling expression, so for 200 people we have 400 pictures.

4.1 Data Preprocessing

Our data are various in size and pixels. It is vital for us to preprocess the data in order to maximize reliable outcome after training. Once we download all the data and put it in the same folder, We start shuffling and normalizing the data. We use cv2 model to do most of image processing including reading the image and resizing the image to 128×128 for every image. We further convert the image to RGB format and normalized the pixel value to be between 0 and 1 with 3 channels. After normalizing data, we divide the data into 24 batch_images and save it as .h5 files. This is because we do not want to load all the data to the platform at once sometimes. Small batch images are more flexible. We will upload the files to google drive we can retrieve data from google drive to colab platform once we mount the folder where the files are in colab. We normalize the data so we have consistent data of the same size, same scale of distribution of pixels which made it easier for us to propagate the data through the pre-defined neural network and the outcome will be less affected by extreme outliers.

4.2 Image Denoising

Since the UTKFace data are collecting from internet without further refinement, some of images are inevitable high in noise and fuzzy. Therefore, we perform image denoising especially for our training data by using openCV api cv.fastNlMeansDenoisingColored(). One of the argument h (parameters) represents the filter strength. We can try different h-value and and windows size and see what's the effect on those images. As you can see, high h-value could lead to less noise and the image become sharper, but it also remove many details of the images making the image overall less natural. During our deep learning training, I usually set the h-value to be either 2 or 3 which can still retain majority of the details of the images and compromise the noise.

[2]https://fei.edu.br/ cet/facedatabase.html

Figure 2: OpenCV Image Denoising Function Example with Different h. High h-value results in less noise and less details remains

4.3 Ethical Issue Pertaining to the Data

Although the UTKFace dataset cover wide range of skin tones and different ethnic group of the different ages. The images of different skin tones or ethnic group are not equally distributed. Majority of the data are the images of middle-age male Caucasian. Hence, the neural network has more chance to learn the details about the features of male Caucasian. When the decoder generate new images, it will intend to generate male Caucasian. It arise ethnic issue and gender bias against the people with dark skin. Similarly, when we construct the black and white images, the models will colorize the black and white images and output the colored one. One way to tackle this is we unsample black and white images and remove the images which has rare skin tones or we can unsample the white male images so the images for different ethnic group could be even.

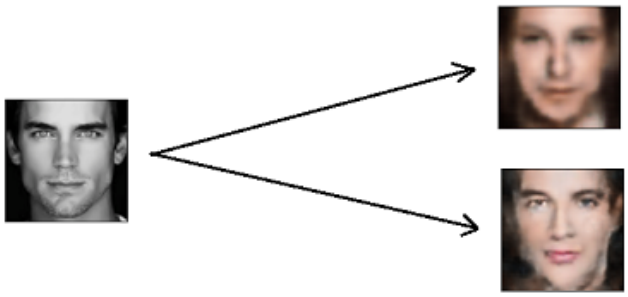

Figure 3: The black and white image on the left is the original images, but somehow our model generated the images with color on the right

5 Design Specification

5.1 First Stage

The first stage has two component for preparation of the partial corrupted images I_m and complement images I_m. It start by checking whether the image contains the holes. The definition of holes in this project are the area of the image which are totally white. We check by counting how many pixels are (255, 255, 255). If they do have holes, then it will pass to face construction block and generate the I_c along with the original I_m. On the other hand, if the images do not have holes which are the case when we train the model, it goes to the random mask block to generate I_c and I_m. The structure of the first stage can be viewed below.

This stage is mainly about image processing and collecting information for image completion. We separate this stage from other deep learning module also because it is

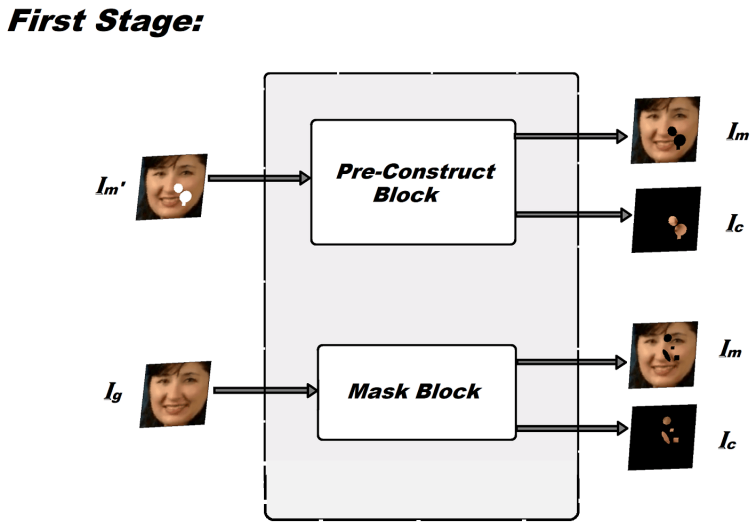

Figure 4: First Stage: Mask Block and Reconstruction Block

reusable and we can change the algorithm without affecting the other stages. It ensure that we are always has I_c which converge the generative phase to the reconstructive phase. Apart from eye detection, I also try face detection and other techniques so we can locate the face accurately but it does not work well. Even for eye detection, the error rate is around 23% using the pre-trained objects. If the system fails to detect eyes we use default value for the center of the face.

5.2 Second Stage

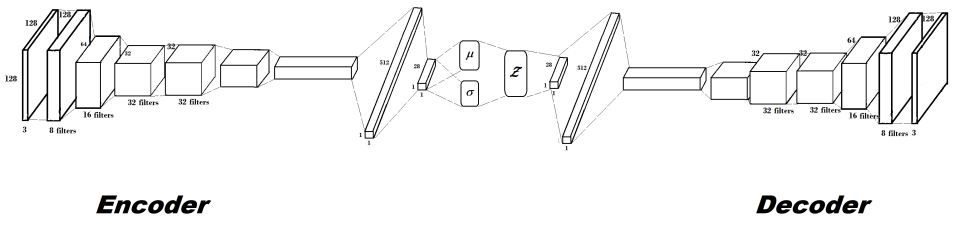

Figure 5: Second Stage: The structures includes Encoder and Decoder

5.2.1 Encoder

Our encoder are composed of three sub-encoder which I denoted as E1, E2 and E3. E1 and E2 come with 3 2D-convolutional hidden layers with filter sizes (8, 8, 16), (16, 16, 32), respectively and kernel size = (3, 3, 3). Kernel size 5 and 7 is also under consideration, however it does not yield better performance in our earlier trials. Since our image size is 128 $\times$ 128 which is not so big and most of the features ares from local, we keep the kernel size 3 for our project. The padding for our layers are the same meaning they were pad the extra pixels so the spatial dimensions would not be affected by the kernel size. I experiment with the stride 1 and 2 for the encoder but it does not have significant impact

on our model in term of performance. Setting stride to 2 will reduce the size of width and length to half of the original size. The encoder will gradually reduce the size layers by layers and make the signals more compact until the signal can pass through the latent space. It is not only reduce the computation time but the size of layers is smaller. E3 is the sub-encoder component that compress the data to the size of the latent dimension. It has two dense layers and output the latent variable z that are sample from the distribution with mean z_μ and the stand deviation z_σ. Combining three sub-encoder components together, we have encoder models that take two input which is I_m and I_c and output the latent variable z, z_μ, z_σ, $eo1$ and $eo2$.

5.2.2 Decoder

The decoder, on the other hand, reverse almost everything the encoder does and learn to construct the data that resemble the I_g. Like encoder, decoder has three sub-decoder components which are D3, D2 and D1. D1 and D2 are corresponding to E1 and E2 and both have similar structure. D1 and D2 have 3 transpose 2D-convolutional layers with filter sizes (32, 16, 16) and (16, 8, 8). They are symmetry of E1 and E2. E3 has two Dense hidden layers. It takes the latent variables that are sampled from the latent space and convert the data to the size of $32 \times 32 \times 128$ after it propagate through the two hidden layers. Sampling from the distribution instead of single point is a trick to make the results meaningful because the latent variable is likely correspond to certain images that are meaningful to us. The decoder take latent variable z, $eo1$ and $eo2$ as input and output the $I_g^{'}$ where the $eo1$ is the neural signal from E1 after it process I_c; $eo2$ is from the result of E2 after I_c propagate through E1 and E2.

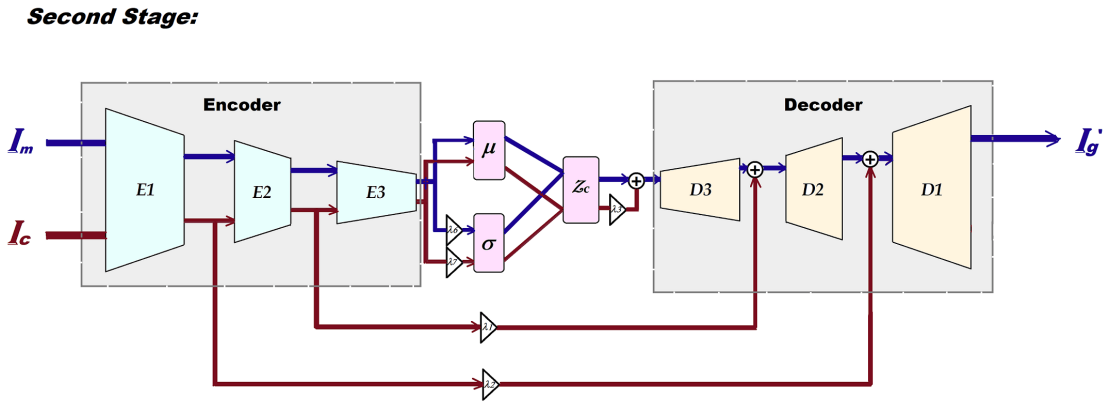

Figure 6: Dual-pipeline CVAE Model Structure

5.2.3 Two-staged Architectures

The overall structures of CVAE can be seen below. It contains Encoder and Decoder to encoder and decode messages. It take three input which are I_m, I_c and I_g and output $I_g^{'}$. The loss function has three components to measure the kl-divergence of z_{I_c} and z_{I_m} and the kl-divergence of the distribution and the construction loss according to Equ. 5. We base on the equation from ($L_{KL} = \Sigma(\sigma^2 + \mu^2 - \log\sigma - 1)$) (Vasilev et al.; 2019) to compute the KL-divergence loss of the distribution. I try the cross entropy, mean square

error (mse) and the combination of the cross entropy and mse for the construction loss between I_g and $I_g^{'}$. Generally, cross entropy shows a slightly better performance in our earlier attempt to build the models but sometimes it is not stable, so I stay with mse. The combination of cross entropy and mse is not ideal either because cross entropy and mse value are not on the same scale. We train the CVAE model with I_m and I_c that are generated by random mask block in every iteration.

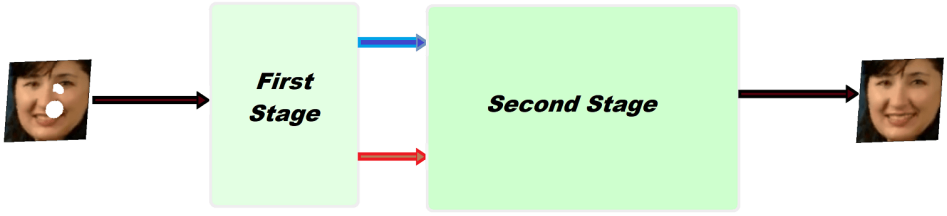

Figure 7: Two Staged Architectures

6 Implementation

Since the google colab limit the RAM to 12.72GB, but our data is close to 5G. So we split the data into 23 batch images otherwise, it will pose a threat to our system. Considering every time we train and test the models, we need approximately $4X$ times the memory to store the data and the caches because we will generate I_m, I_c and $I_g^{'}$ in each iteration. Every time we train the models, we randomly choose 8 to 12 batch images for training and evaluation. Next sub-session summary some deep learning model we use in this project and colab environmental configuration.

6.1 Dual-pipeline Network

When implementing the two-stage architectures, we start by building and testing the components one by one to make sure it works properly. The deep learning models is the core of the structure so that is where we start. I followed the dual-pipeline architecture architectures according to (Zheng et al.; 2019) based on my best understanding of the architectures in the beginning, then I changed the model for simplicity. I experiments the models with different layers and different parameters to see which one perform better.

For the size of the latent variables as an example, more latent variables means more information is able to pass through the latent space so it could contains more details of the original data. You can see it from Fig. 8 that the reconstructed with less size of latent variables appears more fuzzy while the one with large latent variable size preserves more details.

For evaluating the efficiency and the performance of dual-pipeline network, I also built one pipeline network for every the dual-pipeline network as a control group for comparison. I want to assess the performance of our image completion application employed the two-stage architectures. Table 1 display the parameters of one of the dual-pipeline network.

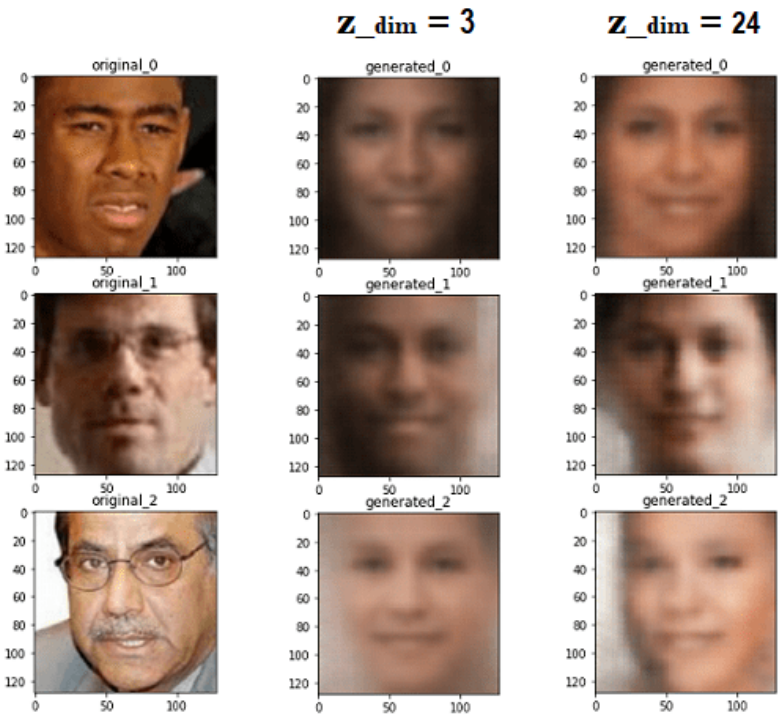

Figure 8: No. of latent variable comparison: left column: original image; middle column: reconstructed images with latent variable of size 3; right column: reconstructed images with latent variable of size 24

λ_1	0.3	z_{dim}	28
λ_2	0.5	latent no	28
λ_3	0.1	kernel	3
λ_4	0.3	epochs	30
λ_5	1.0	batch size	32
λ_6	0/9	img size	$128 \times 128 \times 3$

Table 1: Parameter of of the CVAE Model

6.2 Mask Block and Face Construction Block

Apart from deep learning network, I built the ImageTool class for processing images using the cv2 module. ImageTool offer methods for dealing with images such as convert images from RGB to BGR or mask out an image. It is widely used in both the first and second stage of the architectures.

The face construction block check where is the center of face which is the middle of the two eyes. We use haar cascade algorithm for eye detection and with the pre-trained eye detection objects. All the data are face frontal images. Once we find the center line of the face, we copy the patch from the other side of the face where there is no hole and flip the patch over the center line to form the I_c. If the image is complete and has no hole, then it will be pass to the random mask block in which it generate random mask to mask out the image I_g to form I_c and I_m. The random mask block generate a mask with

holes include the shape of ellipse, circle and rectangles. For simplicity, the holes usually appear on the left-half face so our deep learning module can focus on learning filling the missing pixels on the left-half face. The holes also come in different size every time we generate it but it usually less than one-third of the width of the images.

6.3 Model Training and Testing

Since we distribute the data to the batch images, we use only 4 to 5 batch images for training. It can prevent us from colab crashing and is easier for us to adjust the data size. When we train the model, we input complete image and use the mask block to generate corrupted image I_m and complement image I_g while in the testing phase, we input corrupted image I_m to the face construction block to generate I_c. Both I_m and I_c are the input of dual-pipeline network where the dual-pipeline network will learn to generate the images that are closely related to the original images. We further retrieve patch from the generated images to fill the holes of corrupted images. If the generated images are similar and almost identical to the original images, then we pixels in the generated images corresponding to the area of corrupted images should be the patch that match the holes perfectly like a jigsaw puzzle.

7 Evaluation

Similarity between two images can be measured using structural similarity index measure (SSIM) and peak signal-to-noise ratio (PSNR) or MSE. SSIM and PSNR are also among the most common objective image quality metrics to evaluate the performance of autoencoder. SSIM is the quantity value that show the structure similarity so you can tell how close two images are according to the SSIM. Scikit-image modules offer compare_ssim API to derive the mean SSIM between two images so we use it to evaluate the performance. Generally, when two images are identical then compare_ssim() will give us value one. Apart from SSIM, PSNR represent the difference between maximum power and the noise in the logarithmic scale. We can take the original image I_g as the signal and the difference between I_g and $I_g^{'}$ as the noise and derive the PSNR to see how much loss generated images' quality loss during the compression and generation comparing to the original images. A typical PSNR has the formula:

$$\begin{aligned} PSNR =& 20 \log \frac{MAX_f}{MSE}, where \\ MSE =& \frac{1}{xy} \Sigma [I_{(i,j)} - I^{'}_{(i,j)}] \end{aligned} \tag{6}$$

Typically, we prefer large PSNR and we want the SSIM is close to one as possible because the more similar the two images are the closer the SSIM will get to one. PSNR will also get larger when the generated images is close to the original images. It can be translate that the higher the PSNR and SSIM, the more realistic and closer the generated images are. In my early development of two-stage architectures, I built different models with different parameters to see how good it can achieves in term of SSIM and PSNR. These is done multiple rounds to tune parameters that hopefully will yields better performance. For example, we can fix $\lambda_3 = 0.1$ and $\lambda_4 = 0.3$ and see the impact of different λ_1 and λ_2 on the performance. Here is an example of early experiments results.

case	λ_1	λ_2	PSNR	SSIM
1	0.1	0.1	17.16	0.66
2	0.1	0.6	17.73	0.66
3	0.4	0.4	17.94	0.66
4	0.6	0.1	16.89	0.65

Table 2: PSNR and SSIM for models with λ_3 and λ_4 fixed to 0.1 and 0.3 respectively and with different λ_1, λ_2 values

As you can tell the different λ_1 and λ_2 do not bring significant change to performance but I usually will not pick $\lambda_1 = 0.6$ and $\lambda_2 = 0.1$ because the corresponding PSNR and SSIM is less than the other three cases. These procedure can be carry on and on and try different value for λ_1 to λ_6 in different layer configuration. Eventually we end up using the specification as show in Tab. 1. Besides, in order to evaluate dual-pipeline network, I also built one-pipeline network for each dual-pipeline network as a control group. One-pipeline network is similar to dual-pipeline network besides that it does not take complement image I_c as input so it does not have λ_1, λ_2, λ_3 and λ_4.

Figure 9: The left most column represent the original image, the second left most column represent damaged images and the columns 2 to 4 are the output of different CVAE model. model_1_v5 and model_1_v6 are the one pipeline network and model_2_v5, model_2_v6 are the dual pipeline network

The suffix v5 and v6 delegate models with different configuration. v5 has 6 layers and has 23 latent variable while the v6 has 8 layers in both encoder and decoder and has

latent dimension 28. As you can see from the Fig. 9 that sometimes that hole are filled with pixels that perfectly match the original images but sometimes it don't. It could be the bias of the data that cause the disparity between generated images and the original images.

7.1 Discussion

It is hard to assess the quality of generated images using only human's perception especially when the gap between two images is small. Hence we turn to using SSIM and PSNR along with the visual evaluation. Usually the one-pipeline neural network can outperform dual-pipeline in the beginning or after only a fewer round of training. But as the scale of data get larger and more training, the dual-pipeline neural network can yields around 1 - 5% higher performance than the one-pipeline network. But it is hard to say that the dual-pipeline is always superior than one-pipeline network. In our experiment, one-pipeline network sometimes has better performance than the dual-pipeline especially in the beginning but when they are trains with large-scale data, dual-pipeline might gradually catch up the one-pipeline network or not in term of the performance. Apart from the image generation, some of the pixels might be loss or distorted during the transition between RGB and BGR so these factors also have to be considered in this project.

8 Conclusion and Future Work

We accomplish building the two-stage architectures successfully that is inspired by (Zheng et al.; 2019) (Wu et al.; 2019). In this task, we want to build an image completion application that can reconstruct the downgraded images and fill the hole with the pixels that is content-aware and is semantic consistent. The dual-pipeline network serve pluralistic image completion purpose that is part of our second stage. The first stage, although, seems trivial comparing to the second stage is vital to our project because it make sure the input format always has term I_c and I_m so our reconstructive mode and generative mode are identical in an effort to minimize the gap between training and testing. We spend a lot of time tuning the system and also testing the image processing tools to ensure that our designs is working properly.

However, there still remain a lot of challenge and work that need to be done in the future. I do not dive into partial face image completion and face recognition due to the time constraint in this project. We only use limited information from the right-half face which I believe it cripple the outcome coming from the first stage. On the other hand, our image completion application does not always generate satisfactory output as you can see from Fig. 9. Some generated images seem to have unorganized structures around the hole and sometimes the skin tones does not match well with the original images. It can be improved using edgeConnect proposed by (Nazeri et al.; 2019) or we can develop edge and color network to have smoothly transition between inside and outside the holes. In order to coordinate with edgeConnect, the system can expends to multiple stages to enhance the image completion performance.

Finally, we learn a lot from this project using deep learning and openCV module. We spent time testing the cv2 API for image processing and API from keras module. Although it is still a long way to go, the result of the project shows that our architectures

is effective and two stages can be employed individually to improve the performance and to have semantic consistent outcome.

References

Amara, J., Bouaziz, B. and Algergawy, A. (2017). A deep learning-based approach for banana leaf diseases classification, *Conference in Lecture Notes in Informatics (LNI)* .

Barnes, C., Shechtman, E., Finkelstein, A. and Goldman, D. B. (2009). PatchMatch: A randomized correspondence algorithm for structural image editing, *ACM Transactions on Graphics (Proc. SIGGRAPH)* **28**(3).

Cai, J., Han, H., Shan, S. and Chen, X. (2019). Fcsr-gan: Joint face completion and super-resolution via multi-task learning, *IEEE* .

Deng, Y., Dai, Q. and Zhang, Z. (2011). Graph laplace for occluded face completion and recognition, *IEEE Transactions on Image Processing* **20**.

Deshpande, A., Lu, J., Yeh, M.-C., Chong, M. J. and Forsyth, D. (2017). Learning diverse image colorization, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Elmahmudi, A. and Ugail, H. (2019). Deep face recognition using imperfect facial data, *Future Generation Computer Systems* **99**.
URL: *http://www.sciencedirect.com/science/article/pii/S0167739X18331133*

G., C. and M., T. (2014). *Constrained PatchMatch for Image Completion*, Cham Springer.

Guo, Q., Gao, S., Zhang, X., Yin, Y. and Zhang, C. (2018). Patch-based image inpainting via two-stage low rank approximation, *IEEE* **24**.

Guo, Z., Chen, Z., Yu, T., Chen, J. and Liu, S. (2019). Progressive image inpainting with full-resolution residual network, *ACM International Conference on Multimedia* .

He, K. and Sun, J. (2014). Image completion approaches using the statistics of similar patches, *IEEE* .

Huang, J., Kang, S. B. and Ahuja, N. (2014). Image completion using planar structure guidance, *ACM* **33**(4).

Iizuka, S., Simo-Serra and Ishikawa, H. (2017). Globally and locally consistent image completion, *ACM* **36**(4).

Jiang, T.-X., Huang, T.-Z., Zhao, X.-L. and Ma, T.-H. (2017). Patch-based principal component analysis for face recognition, *Hindawi: Computational Intelligence and Neuroscience* .

Li, J., Wang, N., Zhang, L., Du, B. and Tao, D. (2020). Recurrent feature reasoning for image inpainting, *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, Y., Liu, S., Yang, J. and Yang, M.-H. (2017). Generative face completion, *In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference* pp. 5892–5900.

Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A. and Catanzaro, B. (2018). Image inpainting for irregular holes using partial convolutions, *In Proceedings of the European Conference on Computer Vision (ECCV)* .

Liu, S., Ng, T.-T., Sunkavalli, K., Do, M. N., Shechtman, E. and Carr, N. (2015). Patchmatch-based automatic lattice detection for near-regular textures, *The IEEE International Conference on Computer Vision (ICCV)*.

Nazeri, K., Ng, E., Joseph, T., Qureshi, F. Z. and Ebrahimi, M. (2019). Edgeconnect: Structure guided image inpainting using edge prediction, *The IEEE International Conference on Computer Vision (ICCV) Workshops*, pp. 3265–3274.

Singh, A. K. and Nandi, G. C. (2012). Face recognition using facial symmetry, *CCSEIT 12: Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology* .

Sohn, K., Lee, H. and Yan, X. (2015). Learning structured output representation using deep conditional generative models, *In Advances in Neural Information Processing Systems*, pp. 3483—-3491.

Sun, H., Xu, W., Deng, C. and Tan, Y. (2016). Multi-digit image synthesis using recurrent conditional variational autoencoder, *IEEE* pp. 375–380.

Thomaz, C. E. (2006). Fei face database, `https://fei.edu.br/~cet/facedatabase.html`.

Vasilev, I., Slater, D., Spagacna, G., Roelants, P. and Zocca, V. (2019). *Python Deep Learning: Exploring Deep Learning Techniques and Neural Network Architectures with Pytorch, Keras and Tensorflow*, Packet.
URL: *https://books.google.com.tw/books?id=ESKEDwAAQBAJ*

Wang, X., Tao, Q., Wang, L., Li, D. and Zhang, M. (2015). Deep convolutional architecture for natural image denoising, *IEEE* .

Wei, J., Lu, G., Liu, H. and Yan, J. (2019). Facial image inpainting with deep generative model and patch search using region weight, *IEEE* pp. 67456–67468.

Weng, R., Lu, J. and Tan, Y. (2016). Robust point set matching for partial face recognition, *IEEE Transactions on Image Processing* **25**(3): 1163–1176.

Wenling Shang, K. S. and Tian, Y. (2018). Channel-recurrent autoencoding for image modeling, *IEEE* pp. 1195–1204.

Wu, X., Li, R.-L., Zhang, F.-L., Liu, J.-C., Wang, J., Shamir, A. and Hu, S.-M. (2019). Deep portrait image completion and extrapolation, *IEEE* pp. 2344–2355.

Xie, C., Liu, S., Li, C., Cheng, M.-M., Zuo, W., Liu, X., Wen, S. and Ding, E. (2019). Image inpainting with learnable bidirectional attention maps, *The IEEE International Conference on Computer Vision (ICCV)*.

Yang, Y., Guo, X., Ma, J., Ma, L. and Ling, H. (2019). Lafin: Generative landmark guided face inpainting, *arXiv e-prints* p. arXiv:1911.11394.

Yi, K., Guo, Y., Fan, Y., Hamann, J. and Wang, Y. G. (2020). Cosmovae: Variational autoencoder for cmb image inpainting, *arXiv e-prints* p. arXiv:2001.11651.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X. and Huang, T. S. (2018). Generative image inpainting with contextual attention, *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* .

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X. and Huang, T. S. (2019). Free-form image inpainting with gated convolution, *The IEEE International Conference on Computer Vision (ICCV)*.

Zhang, J., Fan, D.-P., Dai, Y., Anwar, S., Sadat, F. S., Zhang, T. and Barnes, N. (2020). Uc-net: Uncertainty inspired rgb-d saliency detection via conditional variational autoencoders, *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Z., Song, Y. and Qi, H. (2017). Utkface: arge scale face dataset, `https://susanqq.github.io/UTKFace/`.

Zheng, C., Cham, T.-J. and Cai, J. (2019). Pluralistic image completion, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1438–1447.