PHOTOGRAPH ENHANCEMENT
VIA IMITATION-TO-INNOVATION TRAINING SCHEME

By

YI FENG

Thesis Submitted to the University of Ottawa

in Partial Fulfillment of the Requirements for the Degree of

Master of Applied Science in

Electrical and Computer Engineering

UNIVERSITY OF OTTAWA
School of Electrical Engineering and Computer Science

JANUARY 2021

© Yi Feng, Ottawa, Canada, 2021

ABSTRACT

Photographs are acknowledged as a major carrier of visual information, especially in online interactions. The visual quality of photographs significantly influences the efficiency of the daily interaction. Photograph enhancement aims to improve the visual quality of photographs by modifying the pixel values while retaining original semantic information. As photograph retouching software requires operators to take professional training, an automatic photograph-enhancing system can benefit non-expert photographers and save experts from tedious retouching tasks. Modern automatic photograph-enhancing systems utilize convolutional neural networks (CNNs) to approximate the mapping relationship between raw images and manually edited versions.

In this thesis, we present a novel deep learning framework based on an imitation-to-innovation training scheme. Our method integrates a bilateral grid data structure and an adversarial generative network (GAN) to achieve high time-efficiency and appealing retouched output. We also present a bilateral loss function to maintain the piecewise smoothness. Our experimental results demonstrate that our method is capable of recovering vibrant colourization and sharpness from underexposed photographs in microseconds.

ACKNOWLEDGMENT

I would like to give my deepest gratitude to my supervisor Professor Jiying Zhao for his patient guidance and continuous encouragement throughout my years of study. This accomplishment would not have been possible without him. I would also like to offer my best regards and blessing to the colleagues in our laboratory. Their encouragement has supported me to move further.

DEDICATION

This thesis is dedicated to my mother and father. They provide both emotional and financial support to me all the time. We are thousands of miles away from each other, but I can always feel their love.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AdaGrad	Adaptive gradient algorithm
Adam	Adaptive moment estimation algorithm
CAN	Context aggregation network
CNN	Convolutional neural network
DNG	Adobe digital negative (DNG)
ELU	Exponential linear unit
FCN	Fully convolutional networks
GAN	Generative adversarial network
ISP	Specializaed image signal processors
JS	Jensen-Shannon
KL	Kullback-Liebler
LUT	Color lookup table
MSE	Mean absolute error
MSR	Multi-scale Retinex
MSRCP	Multi-scale Retinex with chromaticity preservation
MSRCR	Multi-scale Retinex with color restoration
NAND	NOT-AND
PSNR	Peak signal-to-noise ratio

ReLU	Rectified linear unit
ResNet	Residual network
RMSProp	Root mean square propagation algorithm
SGD	Stochastic gradient decent
Tanh	Hyperbolic Tangent function
SIFT	Scale-invariant feature transform
SSR	Single-scale Retinex
WGAN	Wasserstein GAN
WGAN-GP	Wasserstein GAN with gradient penalty

Chapter 1

Introduction

Photograph enhancement refers to techniques that improve the visual quality of a given image while preserving its semantic information. Images have become significant carriers of information in contemporary online interactions. Users want to obtain images with high perceptual quality to attract more attention and better express their individual viewpoints. Multiple properties contribute to the visual quality of a photograph, such as global tone, illumination, and photographic composition. In addition to illustration purposes, photograph enhancement is also used to improve the accuracy of industrial computer vision tasks, such as object detection and classification. Many techniques are involved in an image enhancement pipeline, such as denoising [10], tone adjustment [11], and contrast enhancement [12]. In this thesis, we mainly focus on the enhancement of the visual quality of underexposed photographs.

1.1 Photograph Enhancement

Almost all images can benefit from certain modifications, such as tone adjustment or contrast enhancement. Some image signal processors are embedded in mobile devices, but the limitation of on-chip buffers restricts the implementations of complex algorithms. Some

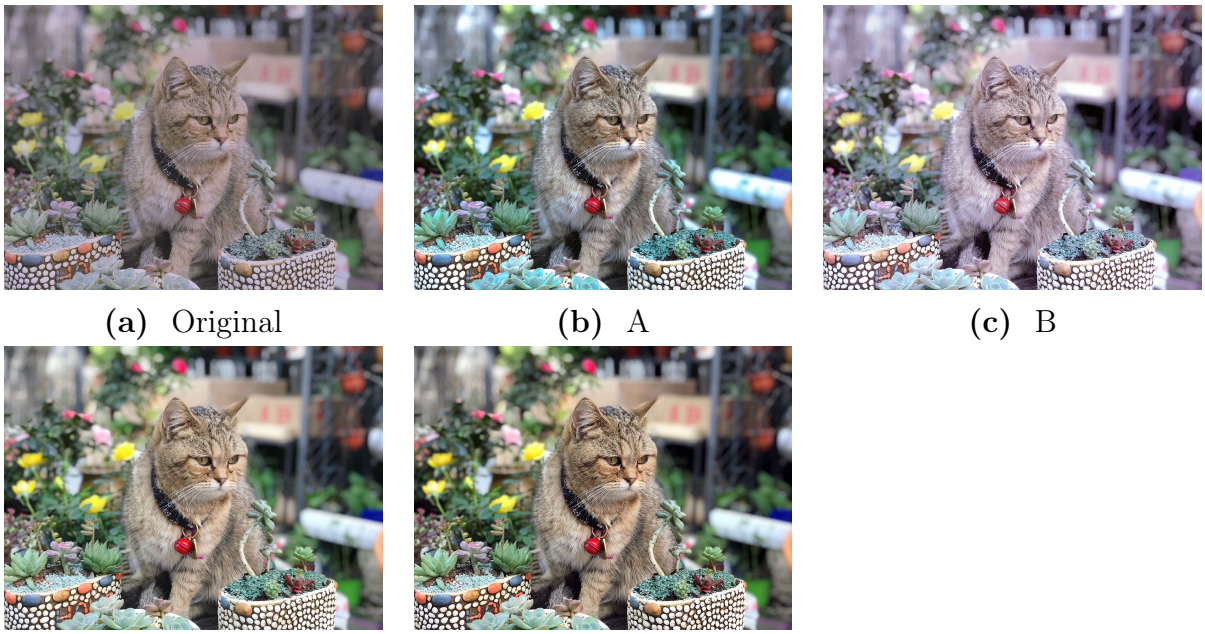
post-processing software provide tools for pixel-level adjustments to maximize the visual attractiveness of an input image. Two widely used professional retouching editors are Adobe Photoshop and Lightroom. These editors provide a rich set of graphic editing tools to satisfy different requirements. However, they require users to receive professional training to master related manipulation skills. In addition, a retouching workflow typically entails tedious and iterative operations regarding global tone and local features. Users often have to test several strategies before achieving satisfaction. Processing a large number of images, like a collection of wedding photos, can be a labour-intensive task even for a professional photography studio.

An automatic enhancement system could save users from lengthy training and heavy workloads. Some mobile applications, such as Meitu and Instagram, provide one-click retouching tools for non-expert photographers. Users can choose from various masks with different styles and can apply the selected mask to their photographs. These masks function via predefined colour lookup tables (LUTs) to modify the global tone. However, only a limited number of predefined styles are available for users, and the use of ill-suited LUT can result in unrealistic colour and destroys the original aura. These mobile applications also provide simplified algorithmic procedures to adjust some graphic parameters such as illumination or sharpness. Users can utilize slider bars to modify the parameters. These one-click mobile applications are convenient, but they are not as powerful as professional software.

Multiple properties, such as illumination, tone, resolution, content, and photographic composition, contribute to the quality of one photo. Designing an automatic image processing system that can handle these high-level features is challenging. Some researchers focus on single property adjustment, such as illumination removal [13], tone adjustment [14], or cropping [15]. However, one photograph usually suffers from multiple imperfections [16], such as low light, overexposure, and haze. These systems usually fail to remedy other kinds of imperfections and have unstable performance. The failure cases indicate that an automatic retouching system is supposed to address all imperfections. Two major obstacles to

practical implementation are given below.

One problem of image retouching is ambiguity. Unlike traditional computer vision tasks, such as automatic detection and classification, aesthetics is a subjective and abstract concept. Photographers have different retouching preferences [14]. Some prefer vibrant and expressive colours, while others incline towards a subdued and natural style. We captured a photo from our daily life and sent this photo to four highly-rating retouching studios in Taobao, which are denoted as A, B, C, D, respectively. The professional retouchers in each studio were asked to edit this photo according to their personal preferences. We obtained four distinct styles, and the outputs are illustrated in Figure 1.1. The subjectivity of the evaluation impedes the design of an automatic enhancement system.



(a) Original **(b)** A **(c)** B

(d) C **(e)** D

Figure 1.1 Different retouching styles on the same image Cat.

Some researchers sacrifice complete automation by requiring users to become involved in the enhancement process, e.g. by selecting a template for style transfer [11]. Some works simplify image enhancement to one-to-one mapping supervised learning. In this case, networks are trained to reproduce the retouching strategy of expert-retouched versions instead

of trying to satisfy different preferences [5, 17]. Some systems are trained to summarize the general features from a collection of high-quality photographs, and apply these features to input images [7, 8].

Another obstacle impeding the development of photograph enhancement is the high time cost. The ever-increasing quality of cameras provides technical support for capturing multi-megapixel photographs, but the computational capacity of mobile devices remains limited. The processing time cost typically grows with the size of the input images, especially when some complex algorithms, like neural networks, get involved.

Gatys *et al.* [18] sacrifices time efficiency for high performance in image reconstruction tasks, which require 1 to 10 minutes to handle a megapixel input, even on a GPU. One solution is to process the downsampled images. The processing algorithms are applied to low-scale input, and the generated output is up-sampled back to the original size as the final result. However, simply applying up-sampling on images inevitably results in blurred outputs. Shan *et al.* [19] proposed a fast algorithm for video and image up-sampling. Cai *et al.* [20] utilized a joint bilateral filter for edge-preserved up-sampling operations. A detailed introduction of the above methods is given in Chapter 3.

1.2 Thesis Contributions

Our proposed method focuses on the real-time enhancement of underexposed photographs. We achieve high-quality performance and low time cost simultaneously by integrating the GAN with the bilateral grid data structure. Like [7, 17, 20], our network generates a single output for each input image. Instead of merely imitating the retouching style demonstrated by dataset producers, our network is also trained to learn from the underlying characteristic of an extensive collection of highly-rated photographs.

The above combination avoids the desaturated colorizations caused by the average effect in supervised learning and the less accuracy of unsupervised learning. A simple combination

sometimes causes posterization and unnatural colour. To maintain a stable system, we present an imitation-to-innovation training scheme and bilateral loss function, which can prevent model collapse in local regions. Our proposed method can produce visually appealing results while preserving the structural and edge information. To our knowledge, our method is the first to achieve real-time image enhancement on megapixel input while not merely imitating the retouch preference of a single expert.

1.3 Thesis Structure

The thesis is organized as follows. Chapter 2 introduces basic concepts of the bilateral grid and neural networks. Chapter 3 discusses several classical image processing algorithms and art-of-the-state photograph enhancement methods based on neural networks. Chapter 4 presents our proposed methods. Chapter 5 presents our experimental results through quantitative evaluation and user study. Chapter 6 gives the conclusion of the whole thesis and introduces our future work.

Chapter 2

Background

2.1 Bilateral Grid

A bilateral grid is a data structure proposed by Chen *et al.* [21] to support rapid edge-preserving image transformation. It is based on a bilateral filter, which assumes images are piecewise-smooth except for edge areas [22]. Because of the high time cost and memory cost when processing multi-megapixel images, some systems apply algorithms on the down-sampled version of input images and up-sample the processed outputs back to full scale. However, up-sampling through an interpolation kernel inevitably results in a blurred output. The bilateral grid data structure provides an efficient solution for rapid manipulation of edge-preserving processing of images. Instead of processing downsampled input images, a bilateral grid enables the algorithm to generate a small set of value maps for pixel-level processing. Neighbouring pixels with similar intensity share information of the same value maps.

2.1.1 Bilateral Filter

The bilateral filter proposed by Tomasi *et al.* [23] is a non-linear smoothing filter which can maintain edge information. The bilateral filter is an improved version of the Gaussian filter. Gaussian filter smooths images by replacing each pixel value with the weighted average

of neighbouring pixel values. The smoothing process is achieved through the convolution between the input image and a kernel. Assume that I represents the input image, and G represents the 2D Gaussian kernel with size $(2k+1) \times (2k+1)$. The output value for the pixel at position (x, y) is obtained through:

$$O(x,y) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G(u,v) I(x+v, y+u) \tag{2.1}$$

where $G(u, v)$ is defined as:

$$G(u,v) = \frac{1}{2\pi\tau^2} e^{-\frac{u^2+v^2}{2\tau^2}}, \tag{2.2}$$

where τ denotes the standard deviation, which determines the degree of smoothing.

The Gaussian filter can reduce noise but also blurs images. The bilateral filter was proposed by Tomasi *et al.* [23] to reduce blurring of edges. The bilateral filter introduces the range Gaussian kernel G_{σ_r} to work with the original spatial Gaussian filter. The range Gaussian kernel is a 1D convolution kernel depending on the intensity difference between neighbouring pixels. Given the input image I and a kernel with size $(2k+1) \times (2k+1)$, the bilateral filter is defined as:

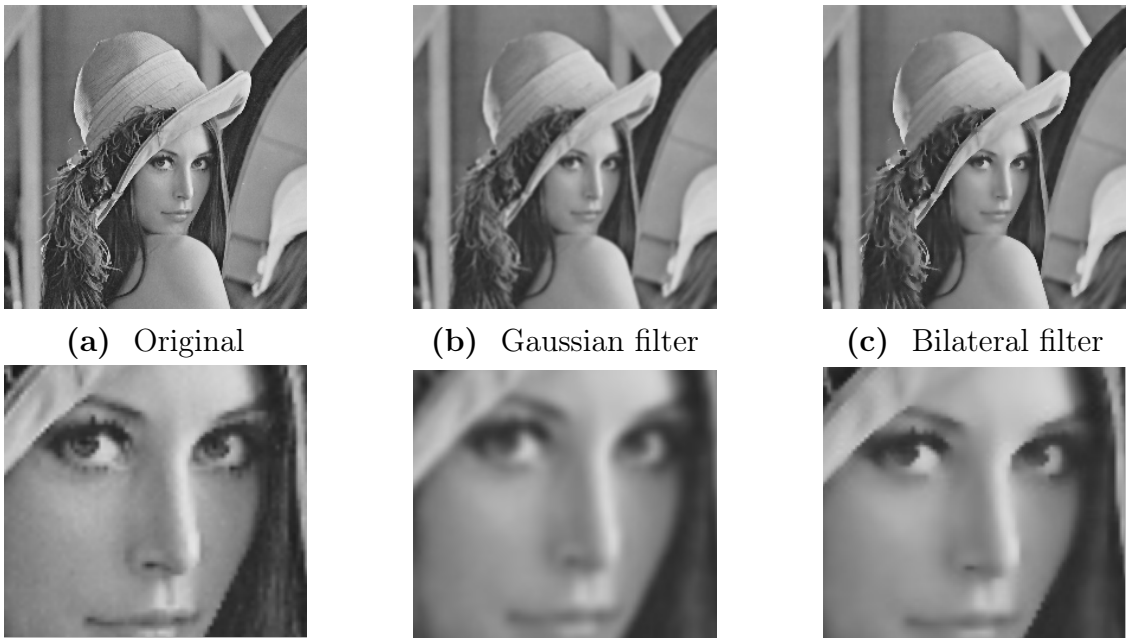
$$B(x,y) = \frac{1}{W_p} \sum_{u=-k}^{k} \sum_{v=-k}^{k} G_{\sigma_s}(u,v) G_{\sigma_r}(I(x,y) - I(x+u, y+v)) I(x,y) \tag{2.3}$$

where G_{σ_s} and G_{σ_r} respectively represent the 2D spatial Gaussian kernel and the 1D range Gaussian kernel, and W_p represents the normalization factor:

$$W_p = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G_{\sigma_s}(u,v) G_{\sigma_r}(I(x,y) - I(x+u, y+v)) \tag{2.4}$$

The spatial Gaussian kernel G_{σ_s} assigns small weights to distant pixels during the convolution. Meanwhile, the range Gaussian kernel G_{σ_r} assign small weights to pixels with a large difference in intensity compared with $I(x, y)$. The introduction of the range Gaussian kernel ensures that the pixels across an edge have a relatively small impact on each other during the weighted average calculation. Thus, the contrast can be better preserved.

Figure 2.1 illustrates the difference between a 5×5 Gaussian filter and a 5×5 bilateral filter. It is evident that the bilateral filter can maintain contrast in edge areas while smoothing images.



(a) Original **(b)** Gaussian filter **(c)** Bilateral filter

(d) Zoom-in of (a) **(e)** Zoom-in of (b) **(f)** Zoom-in of (c)

Figure 2.1 Comparison between the bilateral filter and the Gaussian filter.

The bilateral filter is utilized in various research areas, such as sharpness enhancement [24], image fusion [25], and image dehazing [26]. Although the bilateral filter has demonstrated better performance on image smoothness than other linear filters, its high computational requirement impedes its use in relative practical applications. It usually costs several minutes to process a megapixel image, which is not a favourable property for a real-time image-processing system. Paris *et al.* and Durand *et al.* [27, 28] proposed some enhanced methods for accelerating the convolutional operation. However, these methods usually trade the output quality for time efficiency.

2.1.2 Joint Bilateral Upsampling

With the development of optical sensors in mobile cameras, the size of digital photos has grown to multi-megapixels. Multi-megapixels image imposes high computational and memory load during processing. Algorithms are sometimes applied to the down-sampled version, and the generated output is then up-sampled back to the original size. Upsampling through interpolation kernels will introduce blurring across edges in the final output because of the information loss in the down-sampling process. The joint bilateral filter proposed by Kopf *et al.* [29] address this problem by using the information from full-scale inputs to recover edges in the down-sampling process.

Assume that I represents the original high-resolution input, $S_{\downarrow}$ represents the generated image based on a down-sampled version of I, and d represents the downsampling factor. In the joint bilateral upsampling method, the information in I is utilized to upsample $S_{\downarrow}$ to the original size. The spatial Gaussian kernel is applied to the low-resolution image $S_{\downarrow}$, while the range Gaussian kernel is jointly applied to input I. Let x and y represent the pixel position in I, and $x_{\downarrow}$ and $y_{\downarrow}$ denote the corresponding pixel position in $S_{\downarrow}$. Assume that the spatial kernel size is $(2k+1) \times (2k+1)$. The jointly upsampling process to obtain output $S(x, y)$ at full-scale is defined as:

$$\begin{aligned} S(x,y) &= \frac{1}{W_p} \sum_{u=-k}^{k} \sum_{v=-k}^{k} G_{\sigma_{s_{\downarrow}}}(u_{\downarrow}, v_{\downarrow}) G_{\sigma_r}(I(x,y) - I(x+u, y+v)) S_{\downarrow}(x_{\downarrow}, y_{\downarrow}) \\ W_p &= \sum_{u=-k}^{k} \sum_{v=-k}^{k} G_{\sigma_{s_{\downarrow}}}(u_{\downarrow}, v_{\downarrow}) G_{\sigma_r}(I(x,y) - I(x+u, y+v)) \end{aligned} \tag{2.5}$$

where $x_{\downarrow}$, $y_{\downarrow}$, $u_{\downarrow}$, and $v_{\downarrow}$ are defined as:

$$
\begin{aligned}
x_{\downarrow} &= \left\lfloor \frac{x}{d} \right\rfloor \\
y_{\downarrow} &= \left\lfloor \frac{y}{d} \right\rfloor \\
u_{\downarrow} &= \left\lfloor \frac{u}{d} \right\rfloor \\
v_{\downarrow} &= \left\lfloor \frac{v}{d} \right\rfloor
\end{aligned}
\tag{2.6}
$$

By jointly applying the spatial kernel and range kernel to low resolution image $S_{\downarrow}$ and high-resolution input I, the algorithm can efficiently restore images back to full-size while retaining the adjusted features in the generated image $S_{\downarrow}$ and original edge information in the input image I. Wu *et al.* [9] incorporated the joint bilateral upsampling into the neural network structure to reduce the complexity of neural network models. Nevertheless, the high time cost of bilateral filter convolution aggravates the pressure on computation capability.

2.1.3 Bilateral Grid Data Structure

A new data structure named 'bilateral gird' was proposed by Chen *et al.* [21] as an efficient solution for the joint bilateral upsampling operation. It replaces the bilateral convolution operation by trilinear interpolation in a high-dimensional space. It supports rapid manipulation for some edge-aware operations, such as tone transformation and local histogram equalization.

The 3D bilateral grid data structure can be represented as a 3D grid, as illustrated in Figure 2.2. In this data structure, each node stores a value map for pixel-level manipulation. The total number of value maps is less than the number of pixel values. Neighbouring pixels with similar intensity will share information from the same value maps.

The size of the bilateral grid and the number of value maps are determined by the predefined spatial sampling rate s_s and range sampling rates s_r. Let (H, W) represent the size of the input image, and R represents the dynamic range of input. The size of generated

bilateral grid is determined as $\left[\frac{H}{s_s}, \frac{W}{s_s}, \frac{R}{s_r}\right]$. A lower sampling rate results in a large grid. A large sampling rate reduces memory and time costs, but neighbouring pixels share nearly identical transformation value maps, which sometimes causes posterization. The stored value maps are processed with 2D and 1D Gaussian kernel separately on the range and spatial domains to produce a smooth distribution.

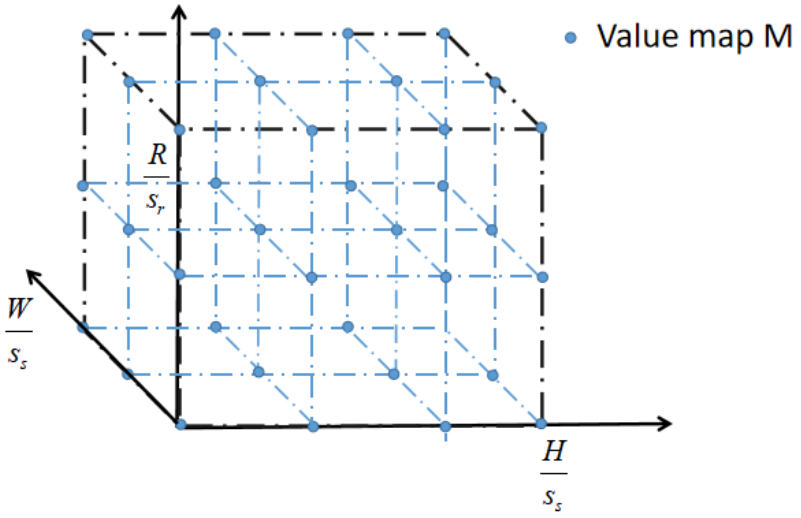


Figure 2.2 The data structure of a bilateral grid.

During the image transformation, the value map for a pixel located at (x_i, y_i) in input I will be extracted at position$\left(\frac{x_i}{s_s}, \frac{y_i}{s_s}, \frac{I(x,y)}{s_r}\right)$ in the grid through trilinear interpolation, which is called 'slicing' in [21]. Trilinear interpolation is the extension of bilinear interpolation. It is used to approximate the value at a point within a 3D grid according to the value on adjacent lattice points.

Through the trilinear interpolation, close pixels with similar intensity share the value maps in the same lattice points, which can maintain the piecewise smoothness except for edge regions. A simplified version based on a 1D image and a 2D bilateral grid is illustrated in Figure 2.3. A 3D bilateral grid data structure is obtained by adding another dimension to images and replacing the bilinear interpolation with trilinear interpolation.

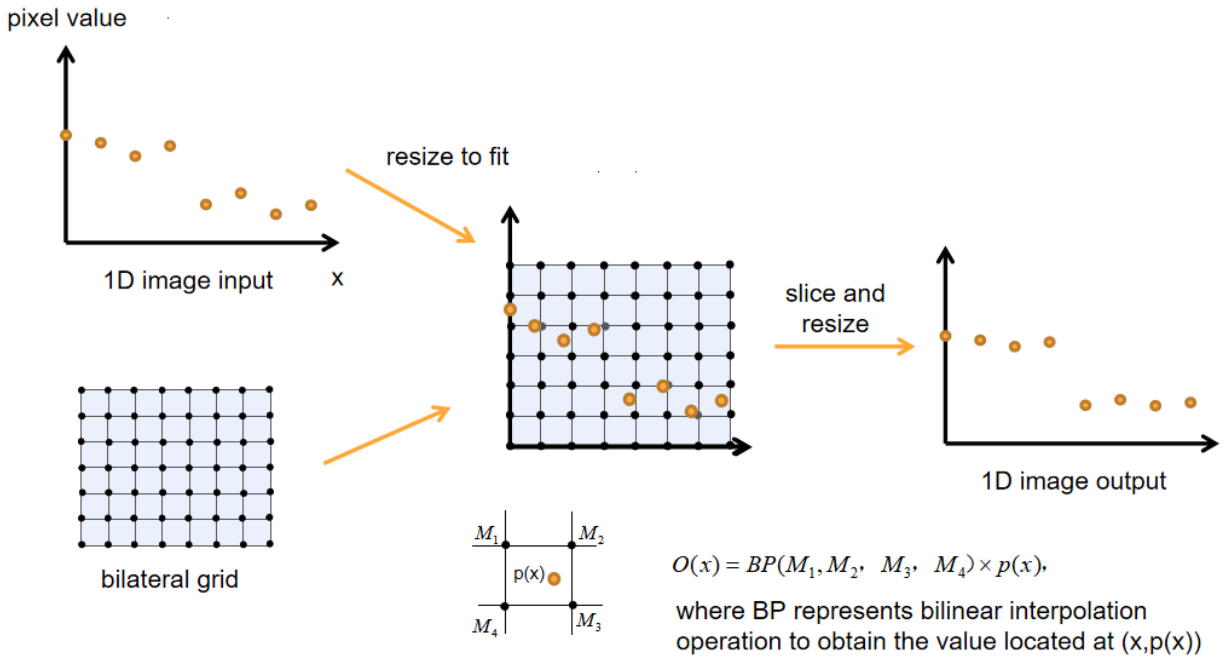


Figure 2.3 The overall pipeline of slice operation on a 1D image and a 2D bilateral grid.

The GPU parallelization technique provides computational support to achieve the slicing operation in real-time. Smoothing a megapixel image via a bilateral grid requires only several microseconds. The main obstacle to the practical implementations of the bilateral grid is the creation of the grid. The image-processing algorithms need to be redesigned to generate a set of value maps stored in a 3D bilateral grid. Chen *et al.* [30] implements local Laplacian filters, style transfer, and matting through the bilateral grid structure. Richardt *et al.* [31] introduced a stereo matching technique based on the bilateral gird. Instead of manually redesigning the processing algorithms, Gharbi *et al.* [17] delegated the generation of value map to neural networks. Neural networks have demonstrated impressive performance in various image transformation tasks, such as image enhancement, colorization and matting [17].

2.2 Neural Networks

Neural networks were first proposed by Warren McCullough and Walter Pitts in [32]. Neural networks are a series of algorithms designed to recognize patterns, detect objects, or implement other manual tasks by imitating the behaviour of biological neuron cells. Neural networks can free humans from the manual design of logistic and mathematics algorithms. With the support of rapid development in the computational capacity of current computers, neural networks have shown vast potential for tackling heavy workloads in real-time applications, such as traffic monitoring and face detection.

2.2.1 Neuron

A neuron is the fundamental component of a neural network. Each neuron processes the received signal and delivers the output to other neurons or to the external environment. All neurons are interconnected to control the data flow from input to output. One basic type of artificial neuron is the perceptron proposed by Rosenblatt *et al* [33]. It has a similar structure to that of biological neurons, as illustrated in Figure 2.4. A neurons receives multiple signals x_i from other cells with different assigned weights w_i. For a biological neuron, when the weighted sum of received stimuli reaches a predefined threshold, the perceptron is activated and transmits the signal into the next stage. The activation function is depicted as:

$$f(x) = \begin{cases} 0 & if \sum_j w_j x_j \leq threshold \\ 1 & if \sum_j w_j x_j \geq threshold \end{cases} \tag{2.7}$$

A threshold-based activation function is a step function. Biological perceptrons generate only binary output, either 0 and 1. In neuron networks, activation function σ and bias b are introduced to bring more complexity into a network. The calculating process is formulated as:

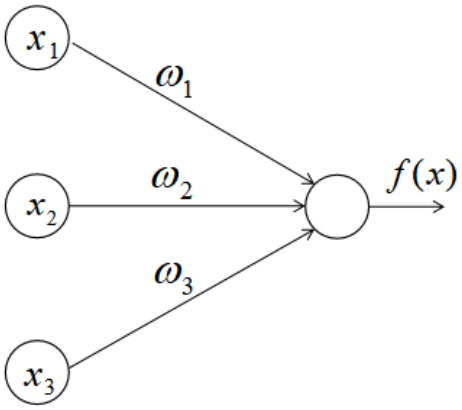


Figure 2.4 The structure of one perceptron.

$$f(x) = \sigma\big(\sum(w_j x_j) + b_j\big) \tag{2.8}$$

2.2.2 Activation Function

Activation functions can restrict the output range, generate numerical values instead of binary values, and introduce non-linearity into the system. They can smooth the gradient curve for both forward propagation and backpropagation processes. There are several widely-used non-linearity activation functions: sigmoid function, hyperbolic tangent function (Tanh), and rectified linear unit (ReLU).

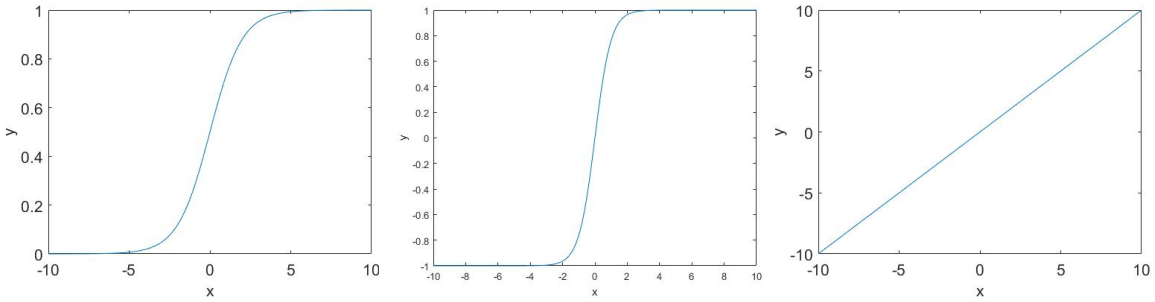
Sigmoid Function:

A sigmoid function is also called a logistic function. The formula is defined as:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.9}$$

Figure 2.5 illustrates the plotted sigmoid function. It has a similar shape to a step function. It restricts the output in the range $(0, 1)$, which efficiently prevents gradient explosion. The output can be used as the predicted probability of classification tasks.

The sigmoid function leads to a vanishing gradient problem and to slow convergence speed. It can be noticed that the function curve at either end has a small gradient. When the gradient is close to 0, the neuron network only receives minimal feedback for neuron weight update, which slows down or stops the training. Despite this defect, the sigmoid function remains popular in neural network design because it can efficiently normalize the outputs.



(a) Sigmoid function **(b)** Tahn function **(c)** Linear function

Figure 2.5 Different activation functions.

Tanh Function:

Tanh function is defined as:

$$tanh(x) = \frac{sinh(x)}{cosh(x)} \tag{2.10}$$

It can be transformed into an exponential expression similar to Equ. (2.9) as:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.11}$$

A comparison between the curves of sigmoid, Tanh and linear function is illustrated in Figure 2.5. It is evident that the Tanh function has similar characteristics to the sigmoid function, but it is zero-centred and has a range between $(-1, 1)$. It has steeper gradients at both ends of the curve. It alleviates the vanishing gradient problem to some degree, but this problem remains.

ReLU:

The mathematical formula of rectified linear unit [34] is:

$$A(x) = \max(0, x) \tag{2.12}$$

The ReLU function curve is shown in Figure 2.6. ReLU sets all negative values to 0 while keeping linear for positive values. This simple operation adds sparsity to activated neurons and introduces non-linearity. The ReLU function is cheaper to compute during training compared to other activation functions. Because ReLU maintains the linearity for all positive values, there is no vanishing gradient problem for ReLU.

However, the ReLU suffers from the dying ReLU problem. When one neuron outputs a negative value, the gradient is 0 during backpropagation as indicated in Figure 2.6. The weight of this neuron will not be updated, and this neuron keeps generating the same value. Thus, the neuron is regarded as 'dying.' In addition, The output range of ReLU is $(0, \infty)$, which may result in the gradient explosion.

Several variants of ReLU were proposed to address the dying ReLU problem. For example, leaky ReLU [35] replaces the negative parts in Equ. (2.12) with a small slope. The function of leaky ReLU is defined as:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha x, & \text{if } x <= 0 \end{cases} \tag{2.13}$$

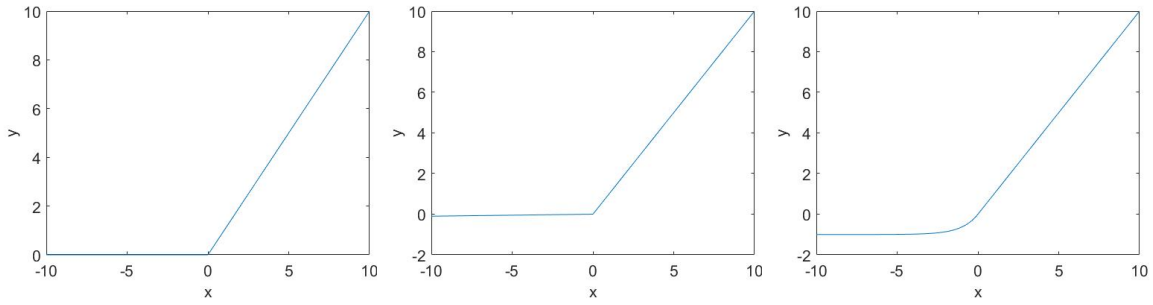
where α denotes a small value, usually between 0.3 and 0.03 [35].

The Exponential Linear Unit (ELU) [36] introduces a non-linear exponential function into for negative values. ELU is defined as:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1), & \text{if } x <= 0 \end{cases} \tag{2.14}$$

where α denotes a small value.

Leaky ReLU and ELU are illustrated in Figure 2.6. A small slope is retained in the negative part so that the gradient descent can provide valid feedback when one neuron generates a negative output.



(a) ReLU function **(b)** LeakyReLU **(c)** ELU

Figure 2.6 Plotted function of ReLU and its variants.

2.2.3 Neural Network

A neural network is composed of a group of connected neurons. In classical feed-forward neural networks, neurons are arranged in layers, and layers are stacked to form the network. A neuron receives signals from neurons in the previous layer and delivers the output to the next layer. Each path is assigned with a weight. A basic neural network with three layers is illustrated in Figure 2.7.

The input layer receives information from the outside world, and the output layer is responsible for generating the final output. The layers between the input layer and the output layer are called hidden layers, which have no connection with the outside. The hidden layer performs nonlinear transformations from the input to the output. By repeatedly calculating the weighted sum through successive layers, the network can produce an approximation of the transformation algorithm between the input and the target.

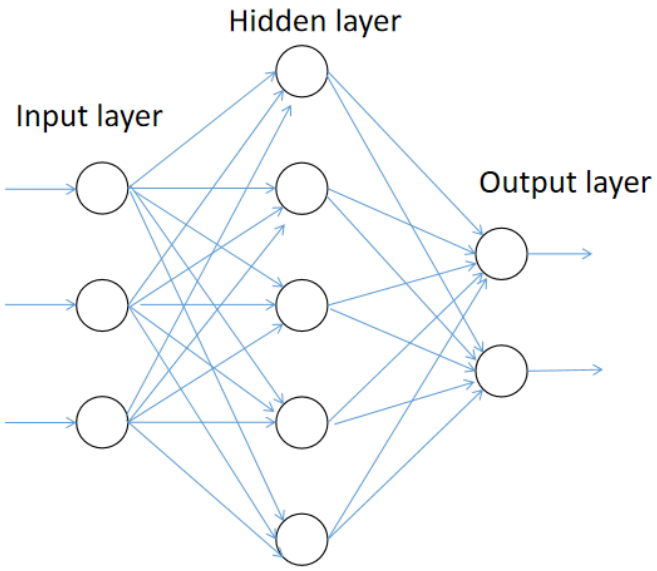


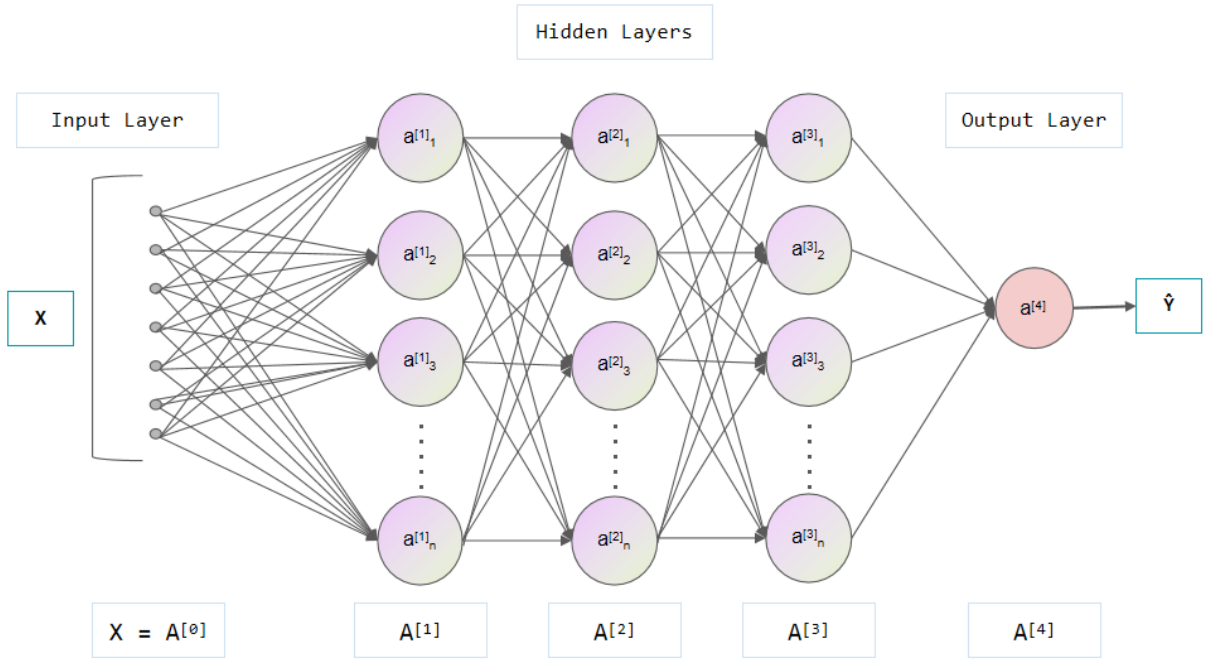
Figure 2.7 3-layer neural network.

2.2.4 Dropout

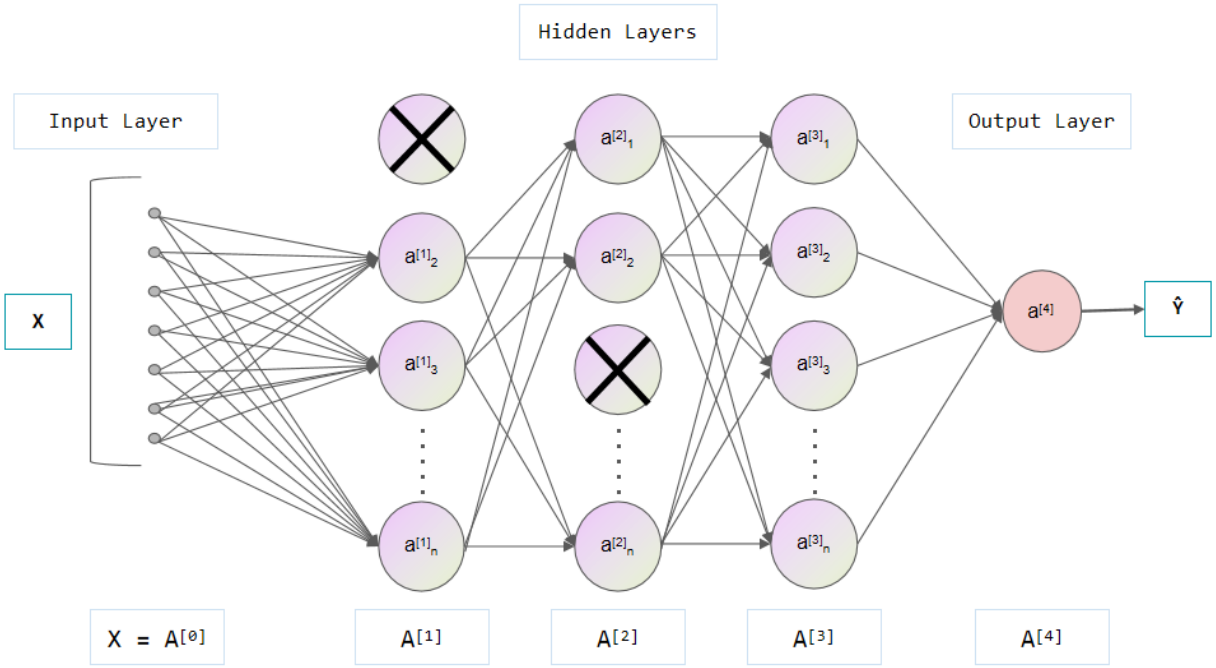
A common problem in network training is over-fitting, which means that the network pays overattention to irrelevant features in the training dataset. Dahl *et al.* [37] introduced the dropout technique to overcome this problem. In the dropout method, random neurons are dropped out during the training process for each example, as illustrated in Figure 2.8. By randomly excluding a certain percentage of neurons from the whole structure, the network becomes more robust to noise and has less chance of being overfitting. In addition, dropping some neurons can also facilitate fast computation.

2.3 Network Training

The ultimate goal of neural network training is to make precise predictions based on input data. There are two types of neural network training, supervised learning and unsupervised learning. Supervised learning provides networks with a labelled dataset, and the network is supposed to process the input and generate output close to the corresponding label. Unsu-



(a) Without Dropout.



(b) With dropout.

Figure 2.8 The dropout function.

pervised learning, in contrast, uses an unlabelled dataset for training. The model is required to learn the underlying features without further information, such as clustering images into different categories without knowing the original labels.

These two training schemes require a loss function to evaluate the performance of the current network. The loss function is responsible for continuously giving feedback to every neuron. The weights and biases are adjusted to minimize the loss function via the gradient descent, which we will introduce in Section 2.3.1.

2.3.1 Loss Function

Loss functions are statistical approximations of the distance between predictions and targets. Training a neural network is to find a set of coefficients that achieve minimal loss. In computer vision tasks, several loss functions are widely used. Mean squared error loss (MSE) and Mean absolute error (MAE) are designed for regression tasks, and cross-entropy loss is common in classification tasks.

In supervised learning, assume that y_p is the prediction from the network, y_t is the target value, and MSE is defined as the average of the squared difference between y_p and y_t. The MSE term is also called L2 loss. Assume there are N samples. The formulation is defined as:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_p - y_t)^2 \tag{2.15}$$

MAE is also called L1 loss. It is similar to MSE. The MAE loss is the average magnitude of the absolute value of errors. Compared to MSE, MAE keeps the error distance linear and is thus more sensitive to outliers. The mathematical formula of MAE is defined as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_p - y_t| \tag{2.16}$$

Cross-entropy loss is commonly used in classification tasks. For classification tasks with multiple categories, the output is usually denoted in a one-hot format. Assume that N

represents the number of categories. A classification network generates the possibility p_i that an input belongs to category i, and the ground truth is y_i. The mathematical formulation of the cross-entropy loss is defined as:

$$Cross\ Entropy\ Loss = -\sum_{i=1}^{N} y_i \log(p_i) \tag{2.17}$$

2.3.2 Gradient Descent and Backpropagation

Weights and biases need to be adjusted to minimize the loss function. Manually modifying coefficients is impractical because of the large number of coefficients and the black-box characteristic. Gradient descent is an automatic and iterative rectification process that modifies neuron coefficients to minimize the loss function. Assume that θ represents the set of all weights and biases, and $J(\theta)$ represents the cost function. By modifying the coefficient θ_0 towards the opposite direction of gradients $\bigtriangledown_\theta J(\theta)$, new tweaked parameters θ_1 bring the cost function one step closer to the minimum. The basic formula for gradient descent is defined as:

$$\theta_1 = \theta_0 - \alpha \bigtriangledown_\theta J(\theta) \tag{2.18}$$

The learning rate α determines how quickly a neural network learns to solve a problem. A low learning rate results in slow learning. A high learning rate can speed up the training, but giant steps in each update sometimes cause the loss function to fluctuate around the minimum and never converge. Learning rate decay is a widely used training strategy. The training starts with a high learning rate, and the learning rate is gradually reduced over a fixed number of training epochs. This training strategy can accelerate training while reducing the possibility that a neural network gets stuck in local minimums.

Neural networks are assemblies of neurons. The weights and biases for all neurons need to be adjusted according to their contribution to the final output. However, only neurons in

the output layer are directly connected to the output. The error needs to be back-propagated from the output layer towards the input layer to obtain the error attributable to each neuron in hidden layers. Computation of gradients is achieved through the chain rule [38], in which the partial derivatives of one layer are utilized to calculate the derivatives of the previous layers. Assume that the forward propagation function with two neurons is illustrated in Figure 2.9. Let $z = g(y)$ and $y = f(x)$ represent the calculation functions of two neurons. The derivative of output z regarding x is defined as:

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x} \tag{2.19}$$

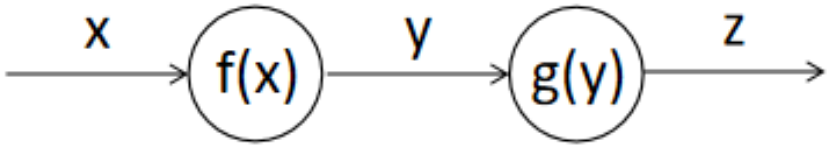


Figure 2.9 Chain rule.

When a neural network is trained with a large dataset, computing gradient descent for every single example has a high time cost and may cause fluctuation. The parameters are usually updated according to the average gradient of a small batch of examples. This scheme is called batch gradient descent [39].

2.3.3 Optimization

Millions of parameters and non-linear activation functions cause a high-dimension non-convex loss function. The fundamental gradient descent update scheme may be stuck at a local minimum. Various optimization strategies for the coefficient update have been proposed to tackle this problem and accelerate the training. These methods are described below.

Stochastic Gradient Descent (SGD)

Instead of updating coefficients according to the average loss of the whole batch, SGD randomly picks up one example from a batch to compute the gradients [40]. By randomly excluding some examples, the SGD works much faster than batch gradient descent, and SGD can import stochasticity into the backpropagation process. Stochasticity helps the algorithm to escape from some local minimums. However, the stochastic update sometimes results in large fluctuations during learning. Thus the network may fail to converge to the global minimum, especially when the training dataset contains some noisy inputs [41].

Mini Batch Gradient Descent

The mini-batch gradient descent scheme [42] contains some advantages of SGD and standard gradient descent. It performs each update on every batch with $n(n > 1)$ examples, which provides more stable convergence compared to SGD. The size of the mini-batch can be adjusted according to different situations and time requirements.

Momentum

Momentum was proposed by [43] to soften the oscillations of SGD. As indicated by its name, momentum optimization retains the momentum of the last update. The current update is influenced by the updating direction of past steps. The mathematical formula of the current update is defined as:

$$\begin{aligned} V_t &= \gamma V_{t-1} + \eta \bigtriangledown_\theta J(\theta) \\ \omega_{t+1} &= \omega_t - V_t \end{aligned} \tag{2.20}$$

where V_{t-1} represents the update in the last step. γ and η represents the weights assigned to the last update and the current gradient.

Adaptive Gradients (AdaGrad)

Neurons in a network have varying levels of importance. The AdaGrad optimization strategy functions by assigning different learning rates to every parameter based on their importance in each step [44]. AdaGrad makes a small update for frequently used parameters and a large modification for unimportant parameters. It helps the network to deal with

sparse data and maximizes the use of all neurons. The update of parameter θ_{t+1} regarding current parameter θ_t is defined as:

$$\begin{aligned} \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} g_t \\ g_t &= \bigtriangledown_\theta J(\theta_t) \end{aligned} \tag{2.21}$$

where θ_t represents the set of current parameters, G_t is a diagonal matrix that sums up the square of past gradients, η denotes the learning rate, and ϵ is a small value to avoid denominator that becomes zero.

Root Mean Square (RMSProp)

RMSprop is an unpublished optimization algorithm. It is similar to adaptive learning models, but it replaces the sum of all gradients squared with an exponentially decaying average [45]. RMSProp can accelerate training and solve the radically diminishing learning rates. The learning rate is determined by the moving average parameter S_{dw}. The formula to update the current coefficient set w is defined as:

$$\begin{aligned} S_{dw} &= \beta S_{dw} + (1 - \beta) dw^2 \\ w &= w - \alpha \frac{dw}{\sqrt{S_{dw}} + \varepsilon} \end{aligned} \tag{2.22}$$

where τ is a small value to avoid the denominator from being zero, α denotes the default learning rate, and β represents the weight assigned to the momentum of last update, which is usually set to 0.9.

Adaptive Moment Estimation (Adam)

Adam uses both the momentum training strategy and an adaptive learning rate [46]. It combines the superiority of the AdaGrad and RMSProp. Assume that m_t represents the weighted average of past gradients, and v_t represents the uncentered variance of past gradients. Adam is defined as:

$$
\begin{aligned}
\theta_{t+1} &= \theta_t - \frac{\eta \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \\
\hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\
\hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\
m_t &= (1 - \beta_1) g_t + \beta_1 m_{t-1} \\
v_t &= (1 - \beta_2) g_t^2 + \beta_2 v_{t-1}
\end{aligned}
\tag{2.23}
$$

where β_1 and β_2 represent the weights assigned to momentum and importance of the neuron, and η represents the default learning rate.

2.4 Convolutional Neural Network

Convolutional neural network was first proposed by LeCun *et al.* [47] for document recognition. It replaces the fully connected structure with convolutional layers. The convolutional layer can reduce the number of parameters in the model and maintain the local information from the previous layer.

2.4.1 Convolutional Layer

A convolutional layer uses a pile of filters to extract features from the upper layers via convolution operation. The filter is also called the kernel. The manipulation of filters is similar to an image filter. The output is calculated through the convolution between the input and the kernel matrix, as illustrated in Figure 2.10. The convolution operation keeps the relative position and captures local features from the previous layer. Filters with different coefficients can perform different operations, such as edge sharpening or image smoothing. By piling the kernels, the convolutional layer can extract high-level features with fewer parameters compared to neural networks. Below are several important settings for a convolutional layer.

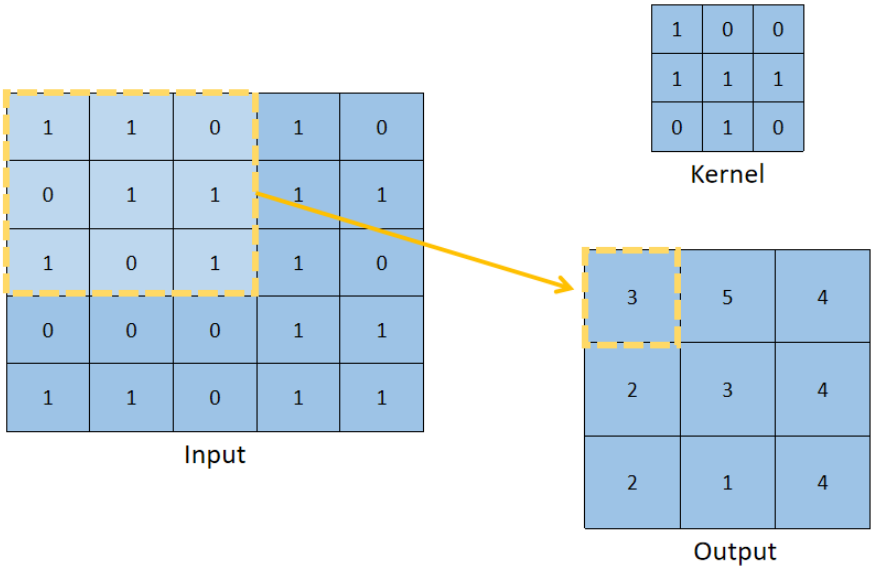


Figure 2.10 An example of the convolutional operation with respect to a single kernel.

Stride

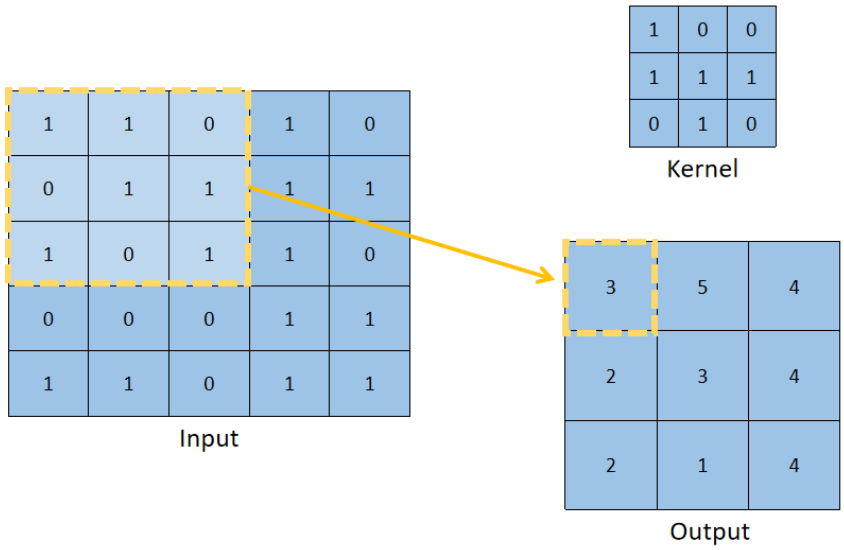
Stride defines how many pixels the kernel shifts for each matrix multiplication. A large stride reduces the size of the feature map, extract high-layer features, and accelerate the subsequent operations. However, less information is preserved with a large stride. A comparison between different strides is illustrated in Figure 2.11.

Padding:

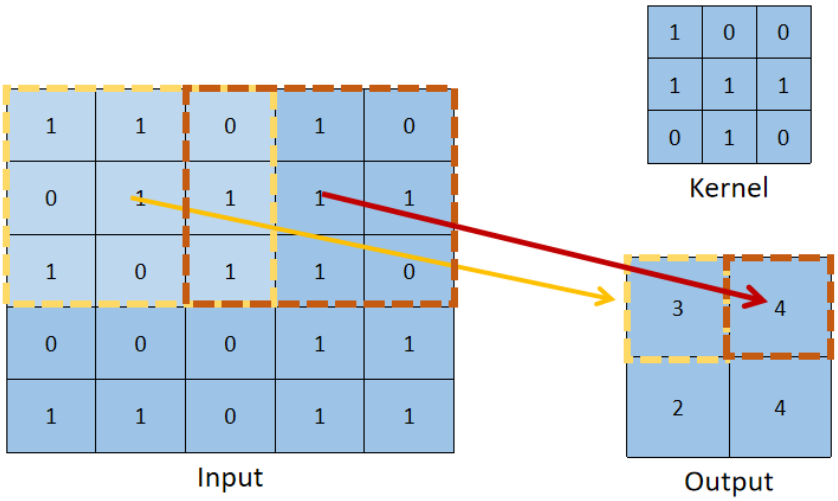
The convolutional operation may cut off some margin information and reduce the output size, which is illustrated in the valid padding example in Figure 2.12. There are two widely used paddings:

1. Zero padding (same padding): It pads the image margin with all zeros. The kernel windows will slip outside the margin of the image.

2. Valid padding: It drops the invalid part and only maintains the valid part. Assumes that the size of the kernel is s. Compared to the output of the zero padding, the valid padding reduces the output size by $s - 1$ on both length and height.



(a) Stride = 1



(b) Stride = 2

Figure 2.11 Comparsion between different strides.

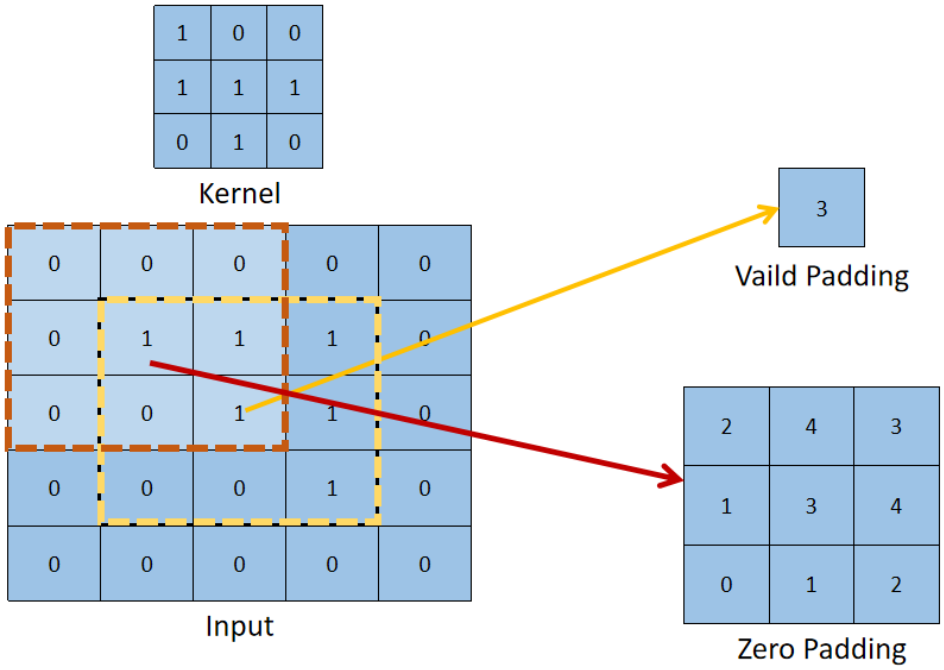


Figure 2.12 Different padding operations.

2.4.2 Complementary Layers

Pooling Layer:

Pooling layers are designed to reduce the spatial dimensionality of a feature map while retaining important information, which is similar to a down-sampling operation [48]. This layer can also be regarded as a convolutional matrix with non-trainable coefficients. There are three popular pooling layers, and their features are illustrated in Figure 2.13.

1. Max pooling: it takes the largest element in the rectified feature map.
2. Average pooling: it calculates the average value of the feature map.
3. Sum pooling: it sums up all values in the feature map.

Fully Connected Layer (FCN):

FCN is used to flatten the output of the previous layer into a single vector. In CNN, FCN was mainly used as the last layer of classification tasks. It can resize the $n \times w \times h$ (w, h represents the width and height of the matrix, n represents the number of feature maps) into

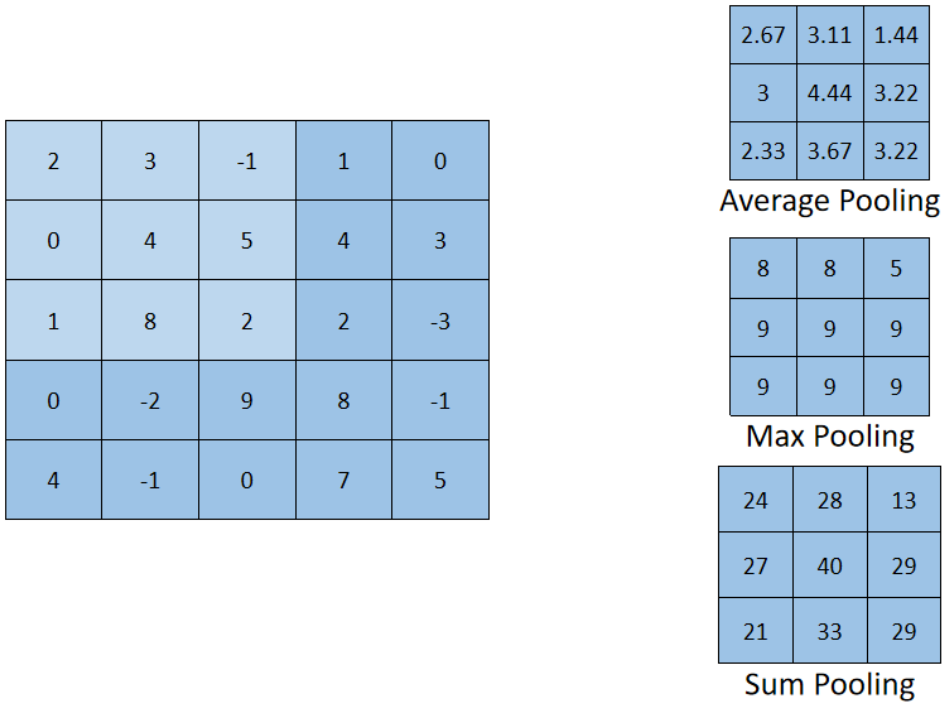


Figure 2.13 Different effects of pooling layers with 3×3 rectified field.

a vector of size $n \times (w \times h)$.

Normalization

The values of inputs to layers can vary from $0 - 1$ or $1 - 1000$. The large gap between different features may cause large fluctuation during training. To speed up learning, scaling the features is necessary.

Batch normalization is a typical normalization method. It is imposed on the whole batch to reduce the covariance shift and align the distribution of different inputs. Supposing μ_β represents the mean value of the mini-batch, τ_β represents the variance of the mini-batch. The weights for standard deviation γ and mean β are trainable during training. The batch normalization is defined as:

$$\hat{x_i} = \frac{x_i - \mu_\beta}{\sqrt{\tau_\beta^2 + \epsilon}}$$
$$y_i = \gamma \hat{x_i} + \beta \tag{2.24}$$

Batch normalization can efficiently reduce overfitting and oscillation. Nevertheless, batch normalization sometimes introduces extra noise into the training process. Instance normalization was devised by Ulyanov *et al.* [49] for style transfer. Instead of normalizing all samples across the batch, instance normalization normalizes spatial dimensions for each individual samples. The modification reduces the noise caused by the distinct data distribution in a dataset.

2.4.3 Weight Initialization

Before training, the weights should be initialized. A suitable initialization strategy can accelerate training and stabilizing the learning process. There are two well-known initialization strategies, He [50] and Xavier [51].

As indicated by Kumar *et al.* in [52], the variance of the layer output reduces during training. Maintaining the variance close to 1 can prevent the vanishing gradient. A model exhibits slow convergence when the variance is smaller than 1. The Xavier initialization randomly assigns weights to each component with the standard normal distribution. Xavier is mainly used for the Tanh activation function. The variance v follows:

$$v^2 = \frac{1}{N} \tag{2.25}$$

He initialization is designed for the ReLU activation layer. As ReLU makes the function non-differentiable at $x = 0$, The variance v follows:

$$v^2 = \frac{2}{N} \tag{2.26}$$

2.5 Evaluation Metrics

Two widely-used metrics, peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) [53], can measure the quality of generated images be comparing to reference images. Given a generated image x and a reference image y, PSNR is defined as:

$$PSNR = 10 \log_{10} \Big(\frac{MAX^2}{MSE(x, y)} \Big), \tag{2.27}$$

where MAX represents the maximum pixel value in x. MSE is the mean squared error.

SSIM is a perception-based model, which identifies the difference between high-level features. SSIM evaluates the difference in three features: luminance, contrast, and structure. The difference of each feature is evaluated through designed comparison functions, and the difference of each property is combined through a combination function to generate the final score.

Assume x and y represent two images to be compared, luminance μ is defined as the mean of all pixel values. The luminance of x is defined as:

$$\mu_x = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{2.28}$$

The comparison function for luminance is defined as:

$$l(x, y) = \frac{2\mu_x \mu_y + C_1}{\mu_x^2 + \mu_x^2 + C_1} \tag{2.29}$$

where C_1 is a small constant in case of denominator being 0.

Contrast τ is calculated through the standard deviation of all pixel values, and the contrast of x is defined as:

$$\tau_x = \Big(\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_x)^2 \Big)^{(\frac{1}{2})} \tag{2.30}$$

The contrast comparison function is defined as:

$$c(x,y) = \frac{2\tau_x\tau_y + C_2}{\tau_x^2 + \tau_x^2 + C_2} \tag{2.31}$$

where C_2 is a small constant in case of denominator being 0.

The structural difference is calculated based on the standard deviation. The difference is defined as:

$$s(x,y) = \frac{\tau_{xy} + C_3}{\tau_x\tau_x + C_2} \tag{2.32}$$

where $\tau(x,y)$ is defined as:

$$\tau_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{2.33}$$

The SSIM score is defined as:

$$SSIM(x,y) = [l(x,y)]^\alpha[c(x,y)]^\beta[s(x,y)]^\gamma \tag{2.34}$$

where α, β, γ denote the importance of each property.

PSNR focuses on the absolute pixel-by-pixel difference. By contrast, SSIM is closer to the human perception model as it compares high-level features. Hore *et al.* [54] indicates that PSNR demonstrates better performance in assessing images polluted by noise, while SSIM better measures the distortion of structural content.

2.6 Generative Adversarial Network

Generative adversarial networks (GAN) are one of the major recent breakthroughs in deep learning. A GAN can automatically create data that follows a specific distribution, such as randomly generating cartoon character painting [1]. GANs have demonstrated vast potential in image generation tasks, such as bedroom photographs creation [55] and face image generation [56]. The main feature of a GAN is that it can be trained in a semi-unsupervised

situation without a paired training dataset. It is capable of automatically summarizing the underlying data distribution from a collection of target items.

An original GAN accepts random noise as input. It has to map the noise in latent space to the generated output, which is usually uncontrollable. The understanding of the high-level information in latent space remains inadequate. The Conditional GAN proposed by Mirza *et al.* [57] allows the received input to be in image or article format. This characteristic supports the conditional GAN to transform input from one domain into another domain. It largely expands applications of GANs. The conditional GAN has been utilized for prediction of face aging [58], automatic makeup application in images [59], and cartoon image generation [60].

2.6.1 Generator and Discriminator

A GAN consists of two networks: a generator and a discriminator. The generator is responsible for producing output following a specific distribution. The discriminator is trained to distinguish between the generated output by the generator and a real sample. The overall structure of GAN is illustrated in Figure 2.14.

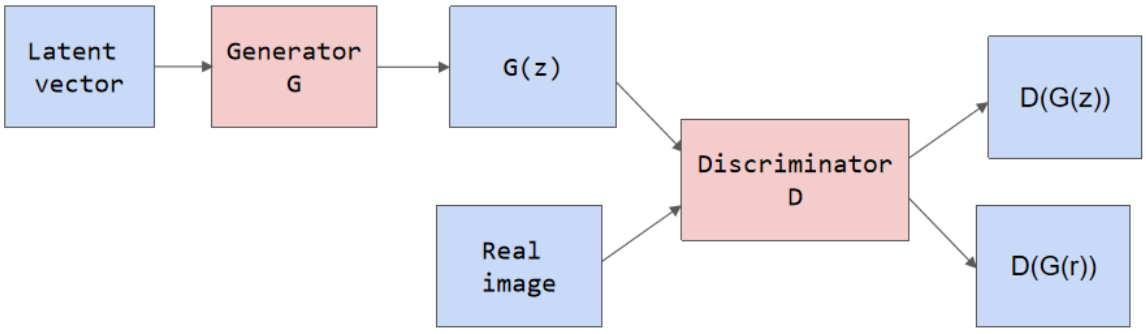


Figure 2.14 Structure of a GAN [1].

The network structure of the generator is similar to the decoder network structure. It receives a latent vector z as input. The input z is processed by multiple transposed convo-

lutional layers (also called deconvolutional layers) and is up-sampled into a full-size image. The randomly-generated latent vector determines the characteristic in generated images. The transformation function can be illustrated as:

$$x = G(z) \tag{2.35}$$

The generator is expected to generate images that have similar features to the given collection. In other words, the generator is responsible for producing counterfeits that are real enough to fool observers. However, as there are no labels for rectifying the behaviour of the generator, theoretically, the generator cannot converge during training.

The discriminator is responsible for judging the quality of generated images. It is trained as a classifier to distinguish between the generated images and real images. The discriminator will receive penalized when mistakenly classifying the input into the wrong categories. Assume that the generated output is $G(z)$ based on noise input z, and $D(x)$ represents the probability that x is classified into the real collection. The objective of the discriminator is to maximize its success rate by giving a high score to x and a low score to $G(z)$. The objective function $V(D)$ to be maximized is defined as:

$$V(D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z}\left\{[\log 1 - D(G(z))]\right\} \tag{2.36}$$

where $p_{data}(x)$ represents the batch of real images, and p_z represents the batch of noise inputs.

The training of a generator and a discriminator goes in parallel. During the training process, the discriminator identifies the features that contribute to a real example, while the goal of the generator is to produce plausible images that are real enough to fool the discriminator. The judgement from the discriminator provides feedback for the performance of the generator. The generator will be heavily penalized when the discriminator rejects its output with a low score. The objective function $V(G)$ to be minimized can be written as:

$$V(G) = \mathbb{E}_{z \sim p_z} \{[\log 1 - D(G(z))]\} \tag{2.37}$$

The discriminator progresses while the generator improves. The training of the two networks can be regarded as a mini-max competition between two players. The final target of GAN training is to obtain a generator that human observers cannot distinguish between a generated image and a real image.

2.6.2 Optimization of GAN

GANs demonstrate an impressive potential for handling automatic image generation problems. However, several potential problems of GANs have been identified:

1. The imbalance between generator and discriminator causes GAN notoriously difficult to train. If the discriminator learns too fast, the generator suffers from the vanishing gradient and may never converge.

2. The generator may collapse and generate only a few identical outputs if there is no additional supervision.

3. The loss function sometimes fails to provide an objective metric to judge the image quality. Hence, humans participation is still required to make the final decision about when to end the training. The timing for manual intervention based on a subjective judgement is sometimes inaccurate.

A classification task is usually easier than a generation task. Hence, during the training of a GAN, a discriminator usually learns faster than a generator. As shown by Arjovsky *et al.* [61], training a GAN is equivalent to minimizing the Jensen-Shannon (JS) divergence and the Kullback-Liebler (KL) divergence between generated images and real images. Assume that p and q represent the data distribution of generated images and real images, respectively. The JS-divergence and KL-divergence are defined as:

$$D_{JS}(p||q) = \frac{1}{2}D_{KL}\left(p||\frac{p+q}{2}\right) + \frac{1}{2}D_{KL}\left(q||\frac{p+q}{2}\right) \tag{2.38}$$

$$D_{KL}(P||Q) = \sum_{x=1}^{N} P(x)\log\frac{P(x)}{Q(x)} \tag{2.39}$$

When the mean of q is close to zero or becomes too large, the gradient of KL and DS vanishes. Therefore, if the discriminator is optimized much faster than the generator, the generator cannot learn from the feedback of the discriminator. As a result, the training sometimes collapses and suffers from the vanishing gradient problem.

One common approach to solve this problem is to enhance the difficulty of the task assigned to the discriminator. However, several methods have been proved to be insufficient to avoid the vanishing gradient problem, such as reducing the discriminator learning rate or adding noise to the feature map in the discriminator [62].

Wasserstein GAN (WGAN) [62] and WGAN-Gradient Penalty (WGAN-GP) [63] avoided KL and JS divergence by using the Wasserstein distance. The Wasserstein distance is the minimum cost of the conversion between two data distributions [64, 65]. Assume the two data distributions are P_r and P_g, the Wasserstein distance between P_r and P_g can be defined as the greatest lower bound of distribution transfer as:

$$W(P_r, P_g) = \underset{\gamma\in\Pi(P_r,P_g)}{inf} \mathbb{E}_{(x,y)\sim\gamma}\left[||x-y||\right] \tag{2.40}$$

where $\Pi(P_r, P_g)$ represents the joint distribution between P_r and P_g.

The original Wasserstein distance is highly intractable. Arjovky *et al.* [62] simplified it as follows:

$$W(P_r, P_g) = \underset{||f||_L\leq 1}{sup} \mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)] \tag{2.41}$$

where *sup* represents the least upper bound. f(x) should follow l1-Lipschitz constraint defined as:

$$|f(x_1) - f(x_2) \leq |x_1 - x_2| \tag{2.42}$$

Compared with KL divergence, the Wasserstein distance has smoother gradients everywhere. As WGAN replaces the sigmoid layer in the output layer with a linear activation function, the range of the output values is $(-\infty, \infty)$. To avoid gradient explosion and to maintain the 1-Lipschitz constraint, WGAN introduces a clipping operation to restrict the weights of the discriminator. The clipping operation keeps all weights in the discriminator network in the range $[-c, c]$, where c represents a tiny value. The experimental results in [62] demonstrated that WGAN maintains stable training even without the batch normalization.

The clipping operation enforces the Lipschitz constraint of discriminator models. However, it puts additional restrictions on the model's complexity as it simply clips all large weights. WGAN-GP replaces the clipping operation with a gradient penalty. Gradient penalty punishes the model if the gradient norm moves away from 1.

Assume D represents the discriminator, and P_g and P_r represent the probability distribution of generated images and real images, respectively. The loss function of WGAN-GP is defined as:

$$L = \mathbb{E}_{\tilde{x} \sim P_g}[D(\tilde{x})] - \mathbb{E}_{x \sim P_r}[D(x)] + \lambda \, \mathbb{E}_{\hat{x} \sim P_\infty}[(|| \bigtriangledown_{\hat{x}} D(\hat{x})||_2 - 1)^2] \tag{2.43}$$

where λ represents the weight assigned to gradient penalty, and $\hat{x}$ is defined as:

$$\hat{x} = t\tilde{x} + (1-t)x \quad with \; 0 < t < 1 \tag{2.44}$$

WGAN-GP efficiently solves the model collapse problem. Furthermore, the generator can still learn from the new feedback when the discriminator was reached its temporary optimal status.

Chapter 3

Literature Review

3.1 Image Enhancement Based on Image Processing Algorithm

Many properties contribute to the aesthetic attractiveness of a photograph, such as the brightness, contrast, tone, and photographic composition. A photograph usually suffers from a diversity of imperfections caused by hardware and environments. Some image enhancement algorithms are embedded in camera devices to process raw photographs. However, processing multi-megapixel raw images in on-chip buffers requires real-time throughput. Hegarty *et al.* and Mullapudi *et al.* [66, 67] focused on the design of specialized image signal processors (ISPs) and optimally scheduling line-buffered pipelines. The designed compiler supports the implementation of high-level image processing codes, such as dynamic range enhancement or motion blurring. Additional knowledge of the corresponding hardware and the compiler is required for the practical implementations.

Due to the limitation of the computation of mobile devices, most image enhancement tasks are transferred to computers. There are some well-known algorithms focusing on the histogram to enhance the image contrast, such as histogram stretching [68] and his-

togram equalization [69]. Examples of histogram equalization and histogram stretching are illustrated in Figure 3.1. Adaptive histogram equalization and contrastive limited adaptive equalization were proposed by Pizer *et al.* [70] to facilitate local contrast adjustment regarding distinct properties in different regions.

However, applications based on the histogram are limited. They usually fail to recover lost information in underexposed images. In addition, simply enhancing contrast does not always produce visually appealing output when a photo has multiple imperfections.

(a) Original **(b)** Histogram equalization [69] **(c)** Histogram stretching [68]

Figure 3.1 Histogram equalization and histogram stretching.

Various methods have been developed to handle different aspects of a photograph. Retinex theory is currently an important algorithm in image enhancement [20, 71]. 'Retinex' is a compound word proposed by Land *et al.* [72] as a combination of 'retina' and 'cortex'. Retinex theory is a perceptual model that mimics the human visual function by decomposing the received visual information into illumination and reflection. Illumination represents the light intensity directed towards the surftace of objects, and reflection represents the light reflected by the surface. Reflection is determined by the physical characteristic of objects. By removing or reducing the illumination, the original colour of objects can be recovered from underexposure or other distortion caused by the illuminating light. In the Retinex theory, a pixel $I(x, y)$ in image I is decomposed into illumination $L(x, y)$ and reflection $R(x, y)$ as follows:

$$I(x,y) = L(x,y) \cdot R(x,y) \tag{3.1}$$

For convenient calculation, the multiplication operation is transferred to addition through logarithmic transformation. The Retinex theory is rewritten as:

$$\log(I(x,y)) = \log(L(x,y)) + \log(R(x,y)) \tag{3.2}$$

Decomposing a single image into illumination and reflection is an ill-posed problem. Researchers on Retinex theory mainly focus on the efficient decomposition of illumination and reflection. Single-scale Retinex (SSR) [73], multi-scale Retinex (MSR) [74], multi-scale Retinex with color restoration (MSRCR) [75] and multi-scale Retinex with chromaticity preservation (MSRCP) [76] are the four classical algorithms for decomposition. These four algorithms are all based on the smooth illumination assumption. After decomposing reflection $R(x,y)$ and illumination $L(x,y)$ from an input image $I(x,y)$, gamma correction is used to reduce the illumination. Gamma correction is defined as:

$$\log(R(x,y)) = \log(I(x,y)) - \log(L(x,y))^{\frac{1}{\gamma}} \tag{3.3}$$

where he parameter γ is set as 2.2 from empirical experience.

Figure 3.2 illustrates several outputs of different decomposition algorithms via gamma correction. As illustrated in Figure 3.2, these algorithms can recover the information in dark regions. MSRCP demonstrates the best performance in recovering colour fidelity. Nevertheless, it has a high compute cost and may introduce redundant artifact halos.

Several variational methods have been proposed to facilitate processing speed and avoid the artifact halo, such as total variation model [77], PED-based algorithm [78], variational framework, and the weighted variational model [79]. A Joint intrinsic-extrinsic prior model [20] has demonstrated superior performance in underexposed region recovery while maintaining the colour information. It can preserve the edge information while achieving low

(a) Original **(b)** SSR [73] **(c)** MSR [74] **(d)** MSRCR [75] **(e)** MSRCP [76]

Figure 3.2 Outputs of different methods based on Retinex theory.

time cost.

The Retinex theory is mainly used for perceptual information reconstruction for under-exposed images. It can be applied to traffic supervisory or other monitoring systems that operate at night. Nevertheless, simply adjusting the illumination sometimes generate flat-looking outputs as it reduces the contrast caused by shadow. In consequence, the results are not usually visually attractive.

Several components contribute to the visual quality of one image, so image enhancement is usually hard to achieve by modifying a single component following a rule-based procedure. Some researchers deal with multiple features transferring features from high-quality images to the given input. The example-based style transfer is achieved through tone transfer [80] or multi-scale contrast transfer [81]. Kang *et al.* [80] provides users with a small collection of photographs C and a visualization interface. Users can retouch given photographs with this interface and edit properties such as temperature, tint, and contrast. The feature vectors and enhancement parameters of each retouched photograph are stored and transfer to input images.

With the assumption that there exists a general strategy to enhance most photographs, Bychkovsky *et al.* [14] collected a dataset named MIT-5k. The dataset contains 5000 raw images and corresponding retouched versions, which were edited by five hired experts. They utilizes regression techniques, such as linear least squares, LASSO, and Gaussian process regression, to extract the general features from high-quality photographs. The image enhance-

ment is achieved by applying extracted features on every pixel through a single luminance remapping curve.

Adjusting photographs based on example images sometimes demonstrates good performance but also demonstrates some problems. Most style-transfer systems are semi-automated. Users are required to retouch photographs manually or provide a collection of high-quality images. Besides, the performance of style transfer depends on the quality and correlation of the provided examples. Transferring the style from the selected examples with unrelated content or distinct colour to the input, such as from portrait images to scenery images, can result in unrealistic tones.

3.2 Image Enhancement Based on Convolutional Neural Network

Neural networks have gained popularity in high-complexity problems. They provide solutions to image reconstruction tasks, such as style transfer[82], image super-resolution [83] and image dehazing [84]. The convolutional neural network was utilized by Lee *et al*. [2] to builds an automatic style transfer system. Instead of requiring users to select style templates as [14], Lee *et al*. collected a large set of high-quality photographs. The network was trained to retrieve style exemplars from the high-quality collection automatically. The retrieved exemplars are supposed to have similar content information to that of the input image. The chrominance distribution and luminance feature were extracted through a multivariate Gaussian and luminance histogram algorithm. The pipeline is illustrated in Figure 3.3. The content-aware selection can retrieve several style exemplars I_h and generate multiple retouching styles for users. However, the style similarity loss function used in [2] failed to provide a fully indicative loss function for network training. Thus, the system sometimes shows unstable performance.

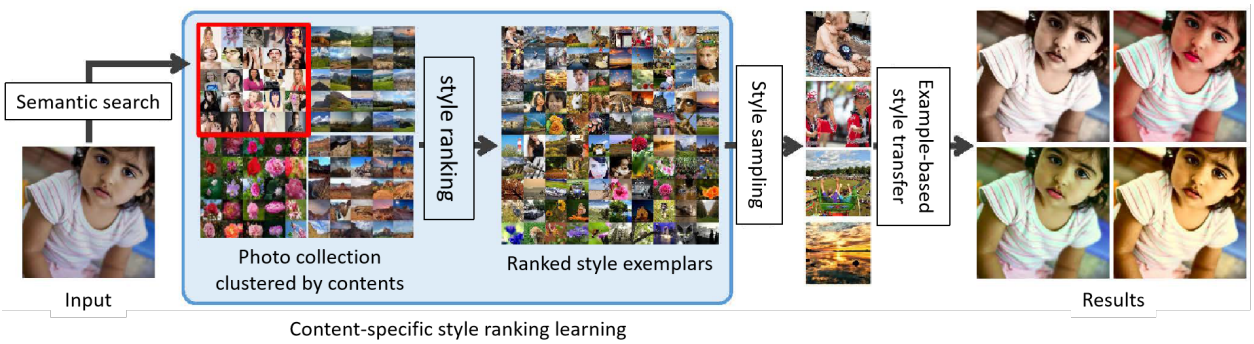


Figure 3.3 The pipeline of content-aware color and tone stylization (adopted from the original paper [2]).

Without style transfer, the main obstacle for applying neural networks to an end-to-end image enhancement system is the multi-modality. Because there is no unique solution for a retouching task, there is also no definite label for supervised learning. The subjective aesthetic preference is still challenging to be transferred to an algorithmic loss function. Some researchers have simplified the retouching task to a single-solution problem. The neural networks are trained to imitate the retouching strategy demonstrated by a single retoucher.

The MIT-5k dataset provides 5000 photographs with corresponding retouched versions, and the retouched photographs can be used as the label for supervised learning. Gharbi *et al.* [17] utilized this dataset and trained the neural network to generate affine transformation matrices for each pixel. The network generated a retouching strategy regarding global and local features. Wang *et al.* [85] adjusted the network structure in [17] by introducing colour loss and smoothness loss. By imposing additional restrictions on illumination smoothness and colour vector, the network can better recover the contrast and colorization of photos.

Ignatov *et al.* [3] utilized the neural network to bridge the gap between photographs captured by mobile devices and DSLP-quality images. Ignatov *et al.* collects a new dataset, which contains 6K photos taken synchronously by a DSLR camera and three mobile devices. An encoder-decoder network is used to reconstruct the low-quality images. The network

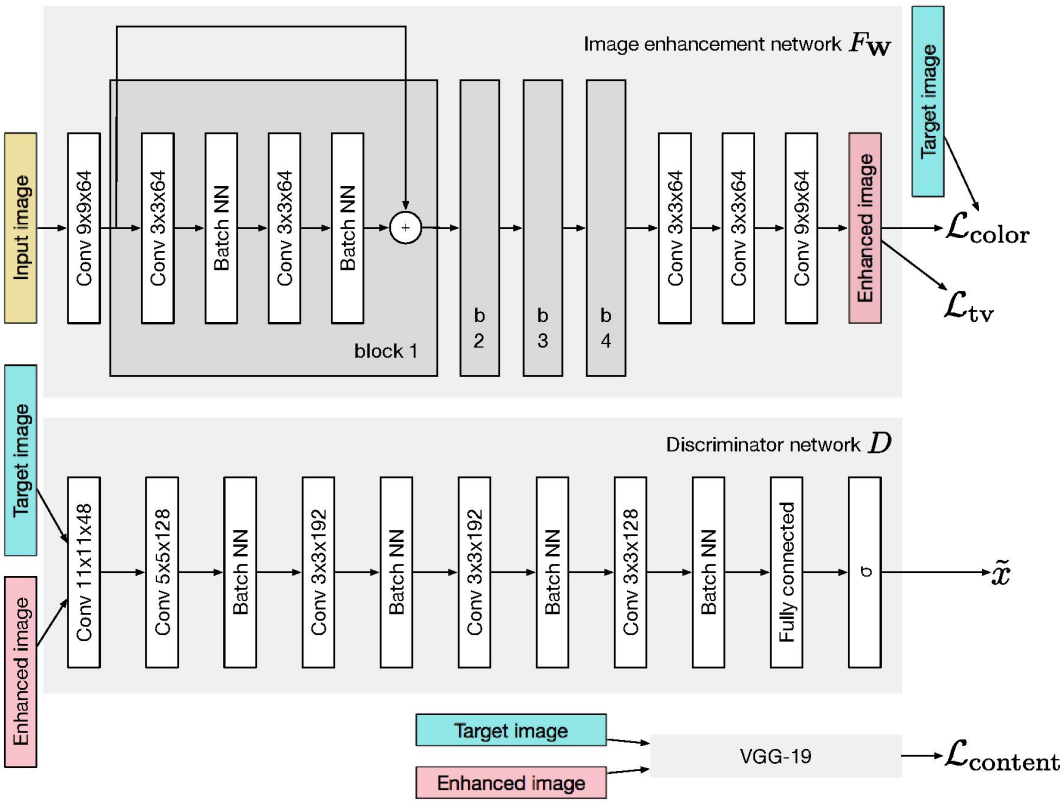


Figure 3.4 The structure of DSLR network (adopted from the original paper [3]).

structure is illustrated in Figure 3.4. Because the synchronously captured images are not perfectly aligned, the MSE loss causes local distortions and blurred patches. Scale-invariant feature transform (SIFT) is utilized to align the content. It builds their loss function by summing up the colour loss, texture loss, content loss, and total variation loss.

Chen *et al.* [4] collected a new dataset which contains 5095 short-exposure images and corresponding long-exposure reference images. Chen *et al.* proposed an end-to-end learning pipeline via U-Net, and the network is trained to handle colour transformation and image enhancement simultaneously. The network structure is illustrated in Figure 3.5. The network operates on raw sensor data, and it can recover colorization information from images with low dynamic range and short exposure time.

Loh *et al.* [5] shifted the focus from enhancing the visual attractiveness to serving industrial applications. It works on the enhancement of imperceptible features to improve the

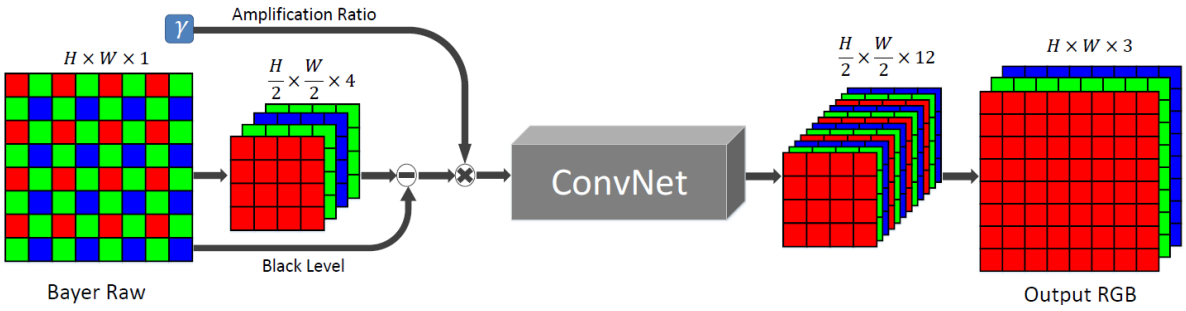


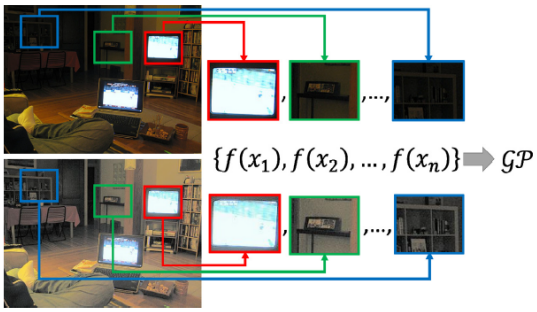
Figure 3.5 The pipeline of the paper 'Learning to see in the dark' (adopted from the original paper [4]).

accuracy of subsequent tasks. Gaussian process regression is used to construct the distribution via a convolutional neural network. The framework is illustrated in Figure 3.6. The Gaussian process transfers the enhancement function into a joint distribution. The network is trained to extract features and perform pixel mapping.

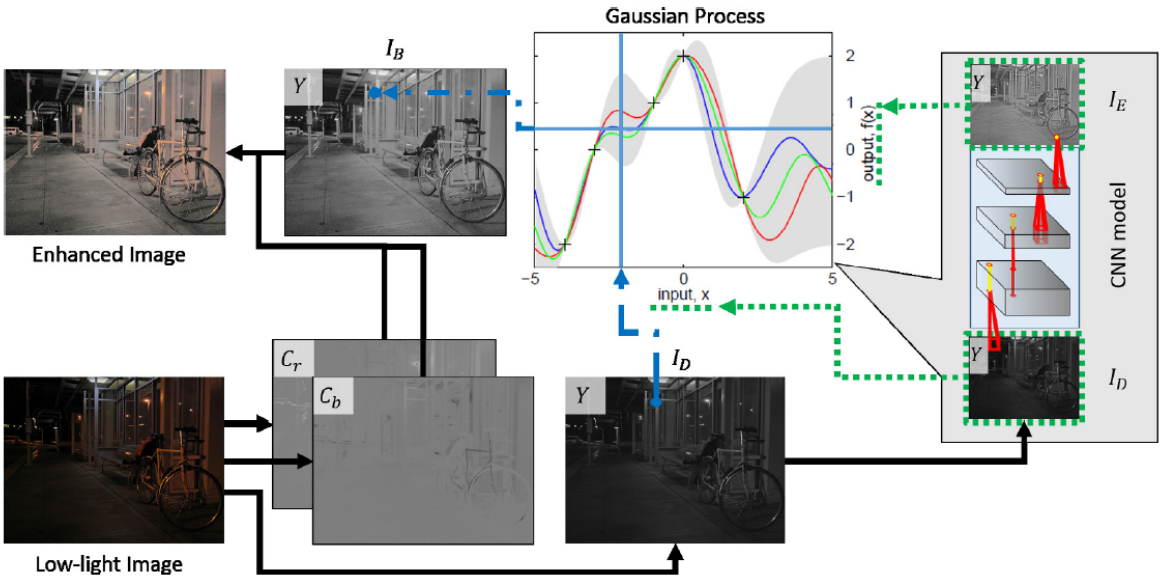
Photograph enhancement is subjective, but these networks trained with single-solution datasets show a high bias towards a specific preference demonstrated by the retoucher. In addition, an object can take a set of distinct tones in several image retouching tasks. For instance, the colour of leaves can vary from dark green to yellowish-green, according to the environment and the mood that a retoucher wants to express. As indicated by Zhang *et al.* in [86], when using Euclidean loss as the loss function, the network takes the mean of all the possible pixel values as an optimal solution. The average effect usually results in greyish and unsaturated outputs.

The development of GAN supports networks to be trained in a semi-supervised way. Training a GAN only requires two image collections without a paired relationship between the samples. In an image enhancement task, the two collections are 1) a collection of raw photographs and 2) a collection of expert-retouched photographs. The generator in a GAN can learn to generate output following the underlying features of high-quality photographs.

Chen *et al.* [6] utilized the encoder-decoder network structure to build an automatic retouching system. This system builds a two-way GAN network structure to maintain the



(a) Gaussian process



(b) The working pipeline

Figure 3.6 The overall framework of features retrieval using Gaussian process (adopted from the original paper [5]).

content and prevent model collapse. The network structure is illustrated in Figure 3.7. The two-way generator can maintain spatial information during the encoding-decoding process. Chen *et al.* also introduced U-Net into the generator network structure to better extract global information.

EnlightenGAN was proposed by Jiang *et al.* [7]. The system was designed to solve low contrast and poor visual perception in underexposed regions. It uses a global-local discriminator structure and a self-regularized attention mechanism to train the GAN. The self-regularized attention mechanism prevents distortion. In addition, it introduces feature

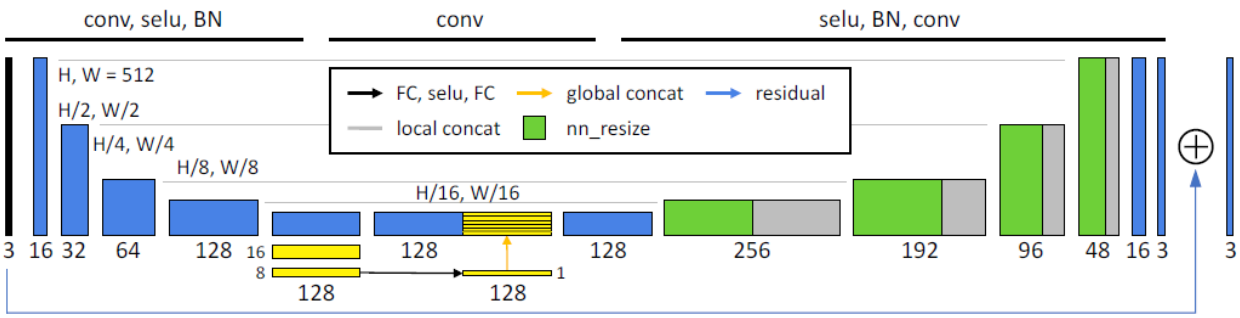
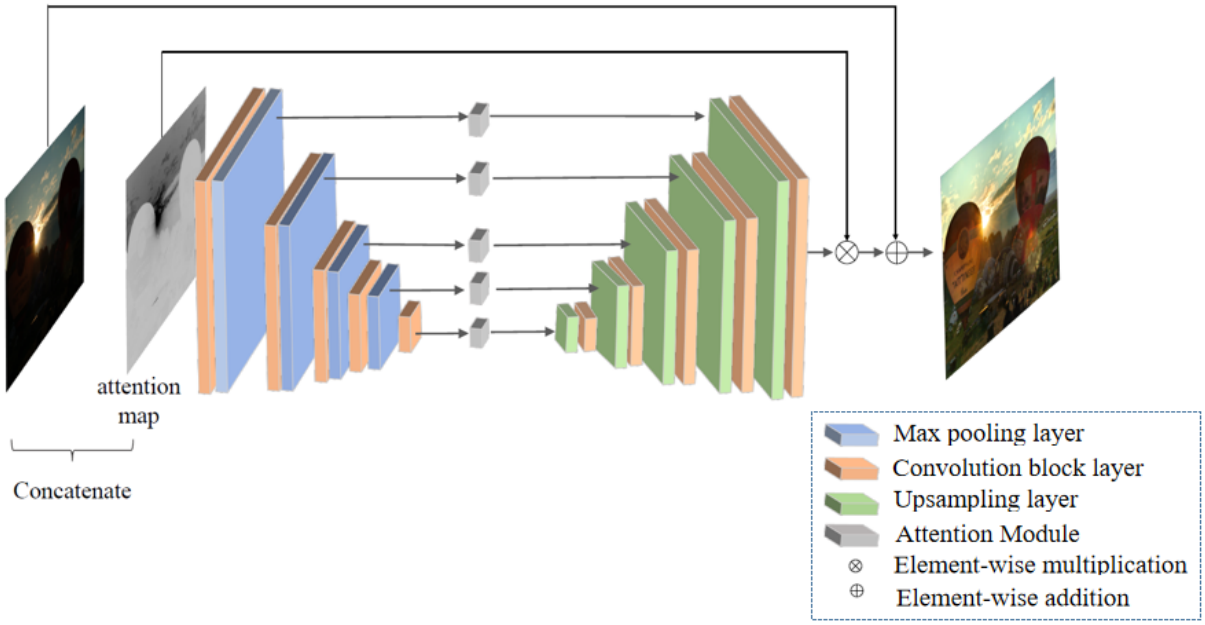


Figure 3.7 The two-way network structure of deep photo enhancer (adopted from the original paper [6]).

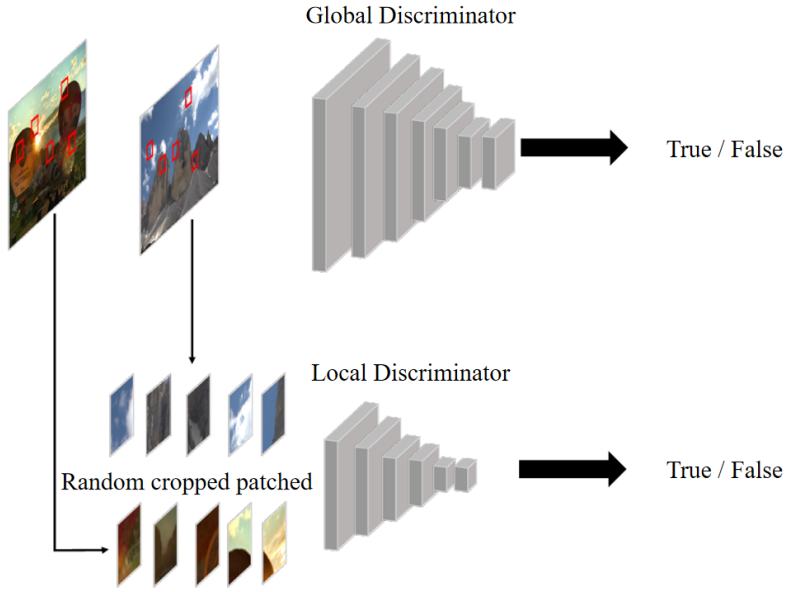
preserving loss to prevent colour distortion. With the support of GAN, the network can be trained with unpaired datasets, which are easier to obtain than paired datasets. The flexibility makes EnlightenGAN adapt easily to other image transformation tasks. The overall network structure of EnlightenGAN is illustrated in Figure 3.8.

A convolution-deconvolution pipeline is usually used for image reconstruction. This network structure usually suffers from high time and space costs, especially when dealing with megapixel input. Users may expect the digital camera to perform both photo capturing and image enhancement, and the expectation places significant pressure on mobile devices. A specially designed algorithm is required to facilitate deep neural networks to operate in real-time.

The convolution-deconvolution pipeline has been discarded in some studies due to its high time cost. Instead of reconstructing the input image through deconvolutional layers, Hu *et al*. [8] designed several image filters with adjustable parameters. These pre-designed filters were kept differentiable and resolution-independent to enable CNN backpropagation. A deep reinforcement network was trained to generate a retouching sequence to apply these filters on input images iteratively. The network structure is illustrated in Figure 3.9. Compared with most black-box models, this work can generate an understandable solution for users to study. The system utilized the discriminator network to evaluate the performance of the reinforcement network. The input images to the reinforcement network were downsampled



(a) Generator



(b) Discriminator

Figure 3.8 The network structure of EnlightGAN (adopted from the original paper [7]).

to 64×64, which significantly reduce the time cost.

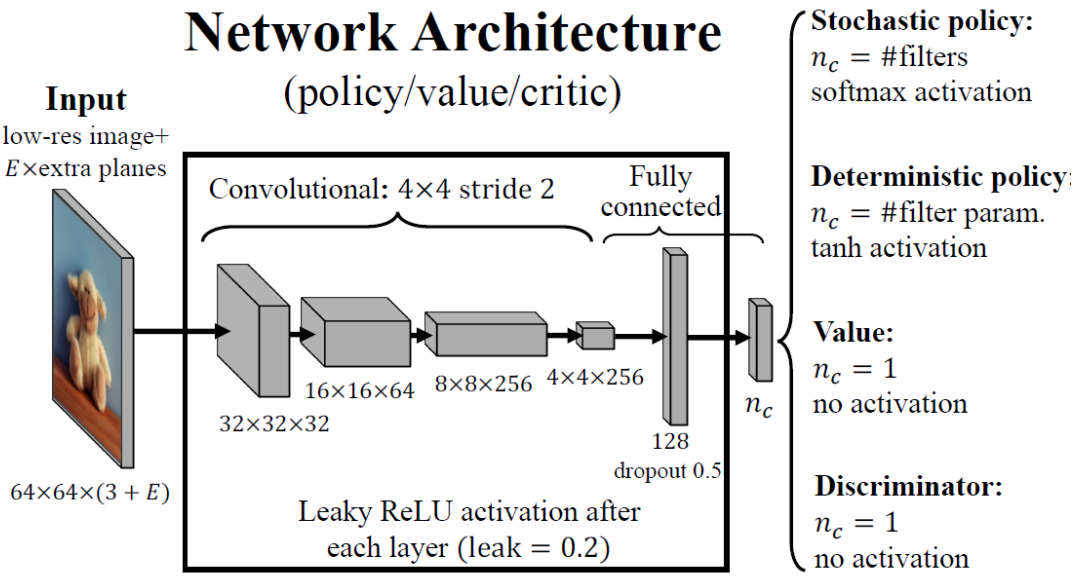


Figure 3.9 The network structure of Whitebox (adopted from the original paper [8]).

Park *et al.* [87] used similar training strategy with [8], but the low-quality dataset was obtained by randomly applying distortion filters to high-quality photographs. It utilizes a Markov decision process to apply global colour adjustment operations on input images iteratively. It extracts colour features and contextual features separately via VGG-16 and CIElab histograms. The agent is trained to perform modification based on the extracted features using the greedy algorithm.

There are some other alternative solutions for building real-time applications. Because of the high time cost of CNN and the limitation of mobile hardware, the computation was transferred to a cloud device in [88]. Wu *et al.* [9] integrated the guided filtering into FCN with an additional guidance map. The guidance map has trainable parameters and provides a task-specific map for joint bilateral up-sampling operation. The computation graph is illustrated as Figure 3.10. The trainable guided filter can be used in different dense pixel-wise image prediction tasks to reduce the time cost.

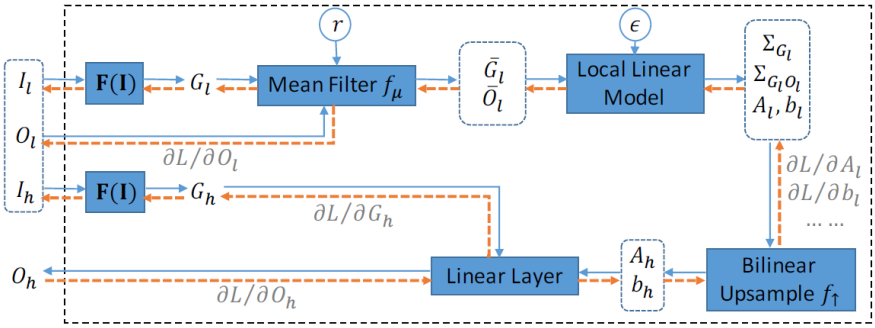


Figure 3.10 The network structure of Trainable Guided Filter (adopted from the original paper [9]).

3.3 Summary

In this chapter, we introduced several photograph enhancing systems based on image processing algorithms and neural networks. The systems based on neural networks can be divided into two categories: 1) systems learning to reproduce the retouching style of a single expert 2) systems learning to summarize a general retouching strategy among a large collection of visually attractive photographs. Both two systems suffer from some potential problems.

We will propose a novel system that can learn from both imitating human retouching strategy and unpaired datasets. The combination can help our system overcome the deficiencies and work in real-time.

Chapter 4

Image Enhancement via GAN Guided Bilateral Grid Network

Image enhancement is typically a challenging task. An image enhancement system should deal with images captured in different environments, satisfy public preferences, and achieve real-time processing. When designing image enhancement via neural networks, generally, three aspects should be considered:

1. A transformation network that can edit the pixel values while maintaining the structural information of images to achieve high visual attraction.

2. An objective loss function to measure the output quality and to provide valid feedback to adjust the weights in the network.

3. A paired dataset containing input-retouched images for supervised learning or a stable unsupervised training scheme.

4.1 Bilateral Grid Generator

The design of a network structure is a long-standing empirical problem, which usually entails a conflict between high quality and low time cost. A widely-used network structure for image

transformation is the encoder-decoder, and this network structure is illustrated in Figure 4.1. The condensed feature maps are extracted from input through several convolutional layers, and the extracted high-level features will be reconstructed to the original size through piles of deconvolutional layers. Encoder-decoder is widely used in image transformation tasks such as [82, 89, 57, 59, 58].

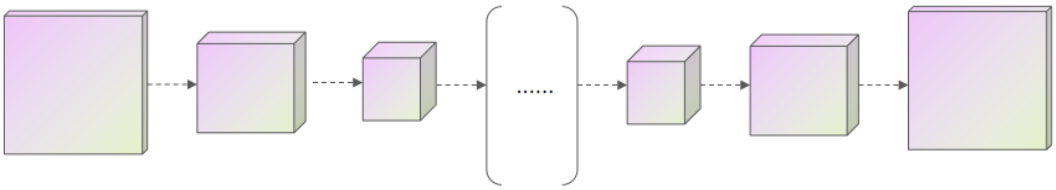


Figure 4.1 The network structure of a classical encoder-decoder network.

Nevertheless, the encoder-decoder network is proved to be inefficient when processing high-resolution image generation tasks. The time cost and memory cost grow with the image size, and multi-megapixel images may cost several seconds to several minutes to finish on a GPU. In addition, it sometimes brings distortion and unnatural colorization into output due to the information loss during the convolutional process. For a photo-realistic photograph processing task, slight distortion results in an unrealistic output.

An encoder-decoder network does not follow the general working pipeline of humans. Retouchers do not decompose one photograph into abstract features and reassemble the high-level information to recreate an image. Human operators tend to determine the retouching scheme regarding the global and local features of input images. Then they apply the retouching strategy on each pixel value. This process retains substantial spatial and structural information of the input image. Following this strategy, instead of directly reconstructing input images, we train the network to generate the retouching strategy and apply the strategy on raw input images.

4.1.1 Bilateral Grid Data Structure

To fully reconstruct the fine-grained details in high-resolution input images, We use the network structure proposed by Gharbi [17] to generate a set of affine transformation matrices and apply the matrices on each pixel in input images. The affine transformation matrix has size 3×4, which is a cascade of 3×3 color transformation matrix M_T and 1×3 bias $[b_r, b_g, b_b]$. Assume that the pixel value is $[r_i, g_i, b_i]$, and the output pixel value $[r_o, g_o, b_o]$ is defined as:

$$[r_o, g_o, b_o] = [r_i, g_i, b_i]M_T + [b_r, b_g, b_b] \tag{4.1}$$

For multi-megapixel images, generating transformation matrices for each pixel value is impractical due to its high time and space cost. Gharbi *et al.* [17] proposed the bilateral grid network to store the affine transformation matrices in a bilateral grid data structure. We used this network structure with some modifications to the layer size and the normalization strategy.

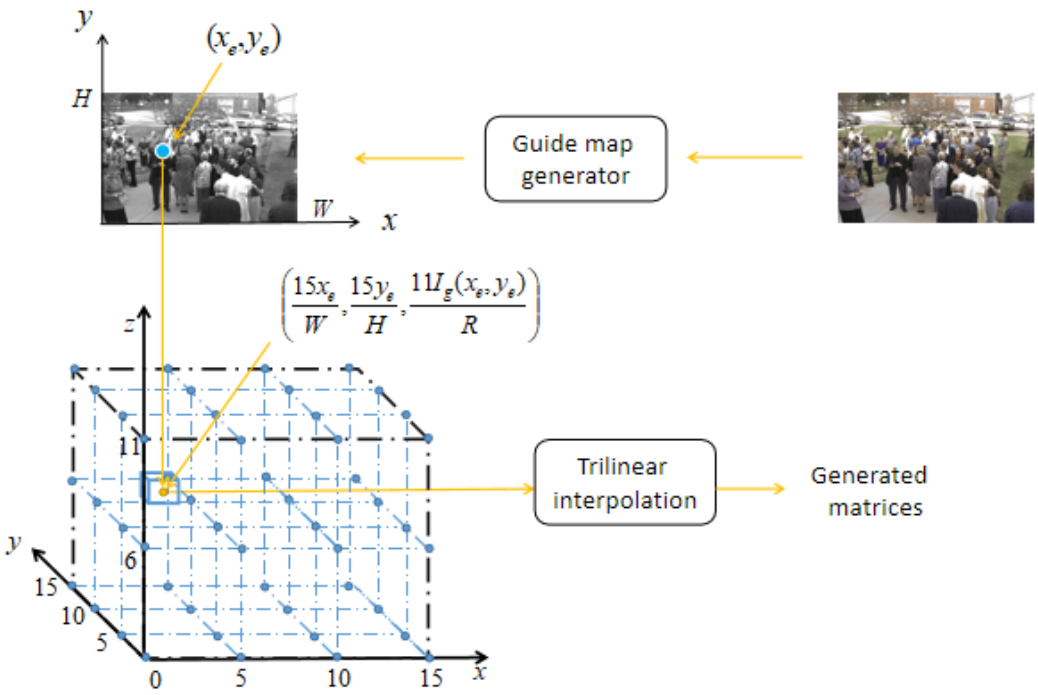


Figure 4.2 3D bilateral grid data structure example.

The bilateral grid can be represented as a 3D grid, as illustrated in Figure 4.2, where each blue dot denotes an affine transformation matrix stored in the grid. The size of our bilateral grid is set as $15 \times 15 \times 11$ in our network, which stores $16 \times 16 \times 12$ transformation matrices. Each pixel in the input image points to a unique location in the grid, and the affine transformation matrix for each pixel is retrieved through trilinear interpolation with respect to adjacent matrices around the location. The retrieval operation is illustrated in Figure 4.2.

Image I with width W and height H first is processed with a guide map generator, which transforms 3-channel images to a 1-channel guide map I_g with the same size as I. The guide map is used to calculate the corresponding location of the affine transformation matrix for each pixel. The detail of the guide map generator is introduced in Section 4.1.3.

In Figure 4.2, each cuboid with eight vertexes (blue dots) has the same size. Each vertex is identified by V_{whr}, where $w, h = 0, 1, \cdots, 15$, and $r = 0, 1, \cdots, 11$. The dynamic range of I_g is represented as R. The affine transformation matrix for pixel at (x_e, y_e) is supposed to be located at $\left(\frac{15x_e}{W}, \frac{15y_e}{H}, \frac{11I_g(x_e,y_e)}{R}\right)$. In other words, the pixel will be enclosed by the cuboid whose leftmost nearest bottom vertex is identified by $\left(\lfloor\frac{15x_e}{W}\rfloor, \lfloor\frac{15y_e}{H}\rfloor, \lfloor\frac{11I_g(x_e,y_e)}{R}\rfloor\right)$, where $\lfloor\cdot\rfloor$ is the floor function. The affine transformation matrix for pixel at (x_e, y_e) is generated by using the trilinear interpolation from the eight matrices at the eight corresponding vertexes of the above-mentioned cuboid. The trilinear interpolation ensures that neighboring pixels with similar intensity share matrices in the same cuboid, which can maintain the piecewise smoothness.

4.1.2 Feature Extraction Network

Our network is trained to generate a set of affine transformation matrices and store the matrices in a bilateral grid data structure. We used the two-path network proposed by Iizuka *et al.* [90]. In [90], two pipelines are designed to extract high-level information and

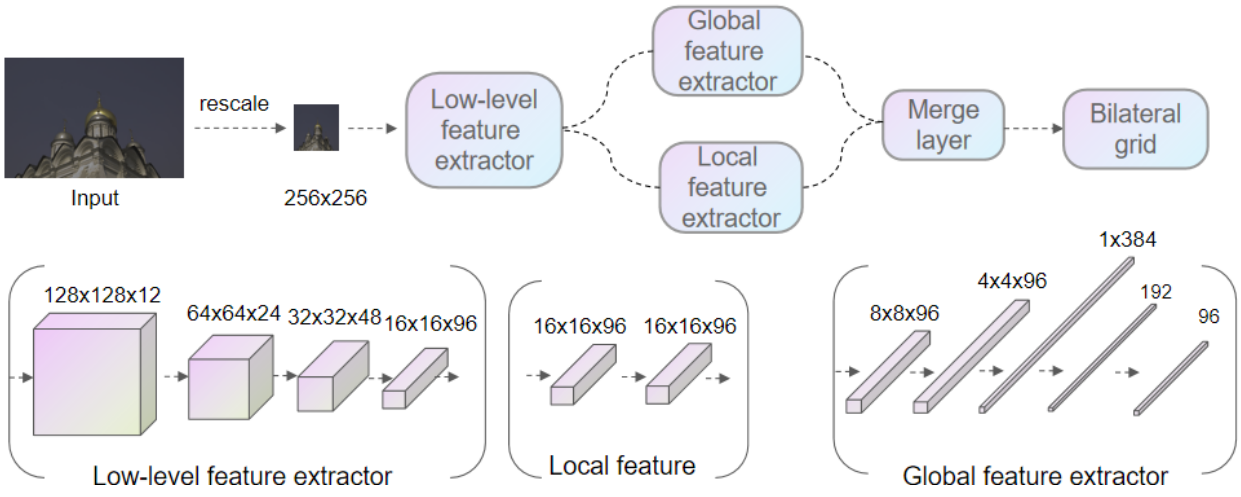


Figure 4.3 3D bilateral grid data structure in our network.

middle-level features separately. Features of different levels are fused to provide holistic guidance for colorization. The network structure of our network is illustrated in Figure 4.3.

The bilateral grid can be constructed based on downsampled images to save time cost. The input image is down-sampled to 256×256. The low-scale image can provide sufficient information for subsequent feature extraction. The low-level features are first extracted through six convolutional layers, as illustrated in Figure 4.4. The extracted low-level features are shared between the global and local extraction paths.

The local feature extractor is trained to generate the modification scheme based on local information. It consists of two convolutional layers. The network structure is illustrated in Figure 4.5a. The size of the final output is $16 \times 16 \times 96$.

The global extractor consists of three fully-connected layers, and the network structure is illustrated in Figure 4.5b. The fully connected layer generates a 1D vector as the global features. The global features can provide holistic information about the semantic understanding of the whole image, such as whether the photograph is taken at night or in the daytime [90]. The extracted global information gives full-scale guidance on image editing.

After the local features and global features have been obtained, they are concatenated into the final output with size $16 \times 16 \times 12 \times 3 \times 4$. It can be viewed as a 3D bilateral grid, where

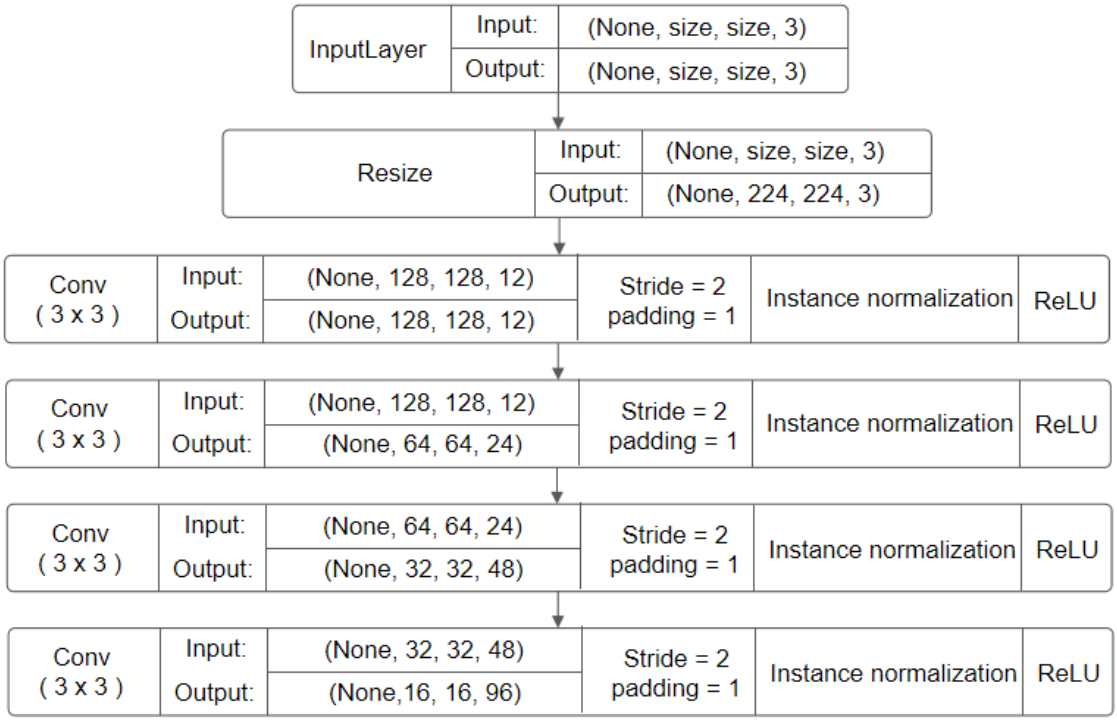
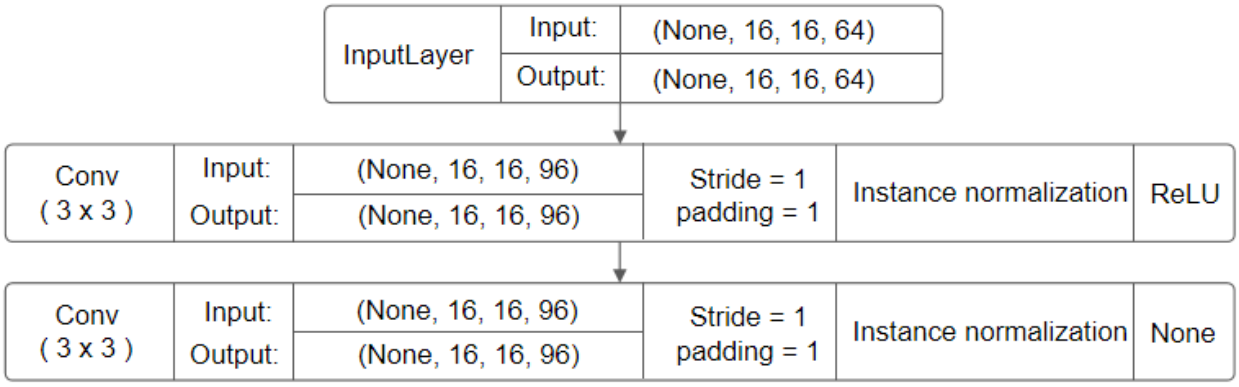


Figure 4.4 Low-level feature extraction.

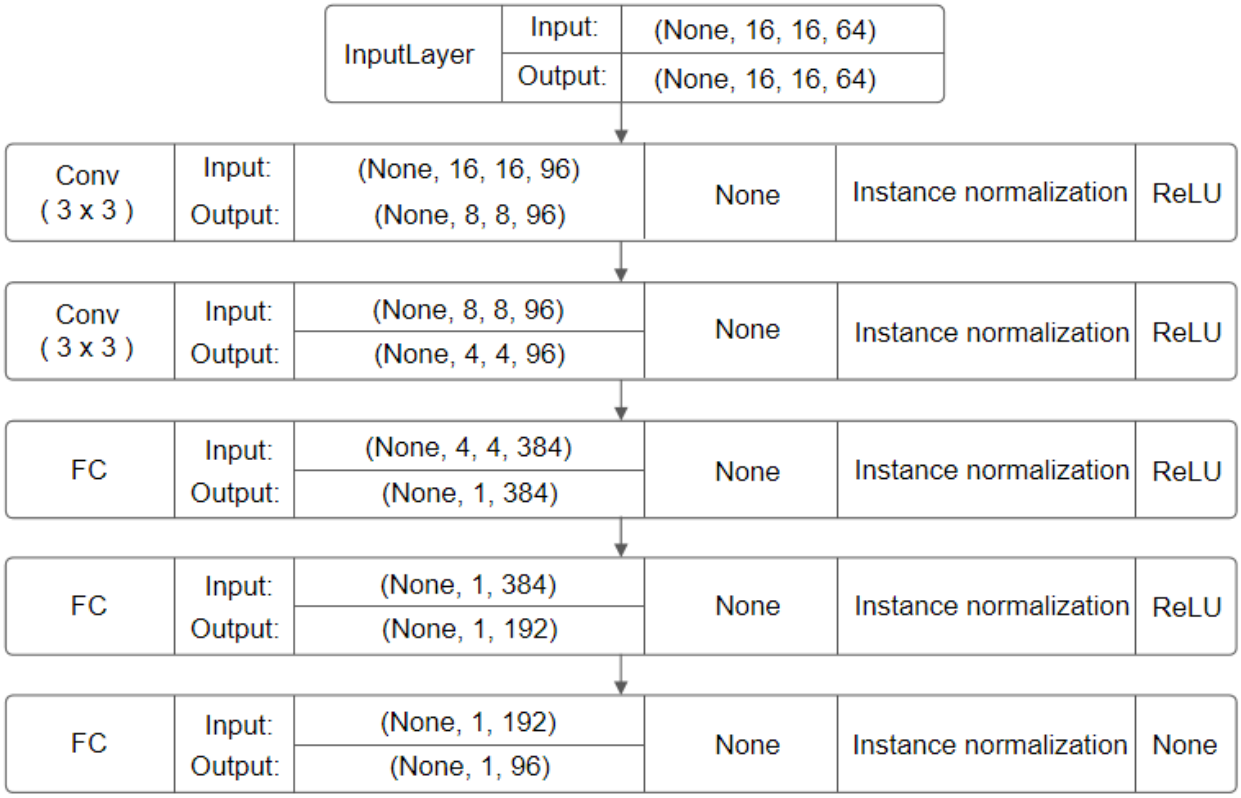
each grid vertex stores a 3×4 transformation matrix. Our experimental results indicated that $16 \times 16 \times 12$ bilateral grid provides sufficient information for image transformation.

4.1.3 Guide Map

Because our work mainly addresses the retouching of underexposed photographs, the network should deal with the ill-posed problem to recover colour in dark areas. The ablation studies in [17, 9] shows that some processing on the input can improve the quality of outputs. The processing is achieved through several convolutional layers instead of an algorithmic procedure. Besides, these convolutional layers can transfer input images into one-channel maps for subsequent retrieving operations, which demonstrate better performance than simply converting RGB images to grey images. In our network, The network for guide map generation consists of two convolutional and one deconvolutional layer. The network structure of our guide map generator is illustrated in Figure 4.6



(a) Local feature extraction.



(b) Global feature extraction.

Figure 4.5 Local and global feature extraction pipeline.

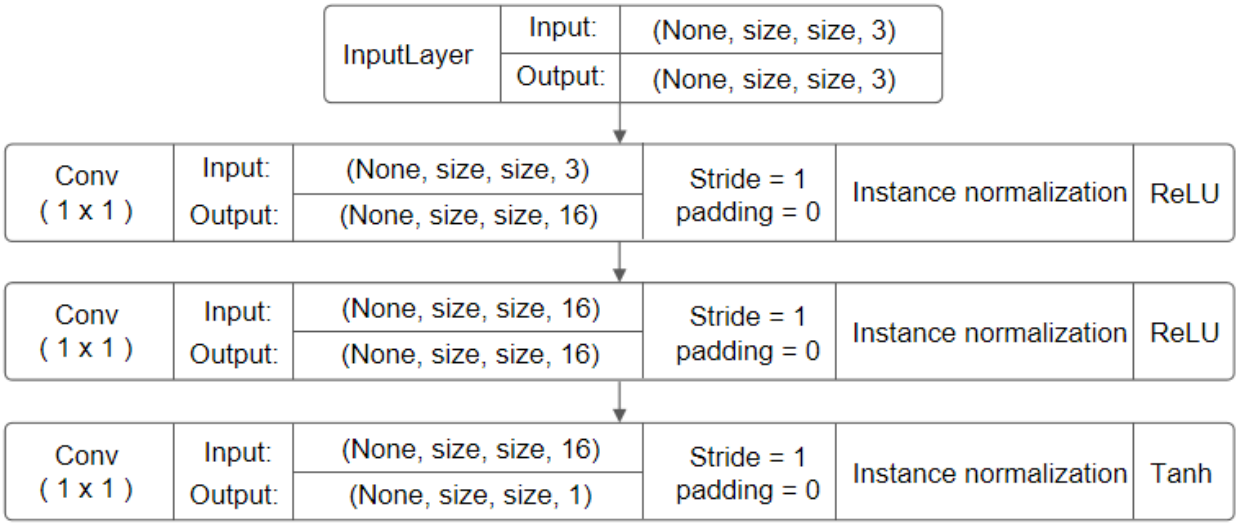


Figure 4.6 The network structure of guide map generation.

4.2 Discriminator

A photograph retouching task has multiple solutions. Hence, loss functions that calculate the pixel-level difference between generated outputs and references, such as MSE, can only work as a simplified evaluation metric. They fail to reflect the retouching quality when receiving a photograph edited with a distinct retouching style compared to the reference image. In addition, as Zhang *et al.* indicated in [86], using MSE to recover colorization information usually results in unsaturated and greyish outputs. A single object can display various colours in different environments. After the iterative training process, the network tends to choose the average values of multiple possibilities. To address this problem, we encourage the network to be 'innovative' and 'brave' to apply vibrant colorization by introducing GAN into our system. The bilateral grid network will receive a binary evaluation from MSE and a discriminator like Pix2pix [89].

GAN is a semi-supervised learning model used for automatic data-driven generations. GAN consists of two networks, a generator and a discriminator. The goal of the generator is to learn from a collection of samples and produce counterfeits. The discriminator is trained

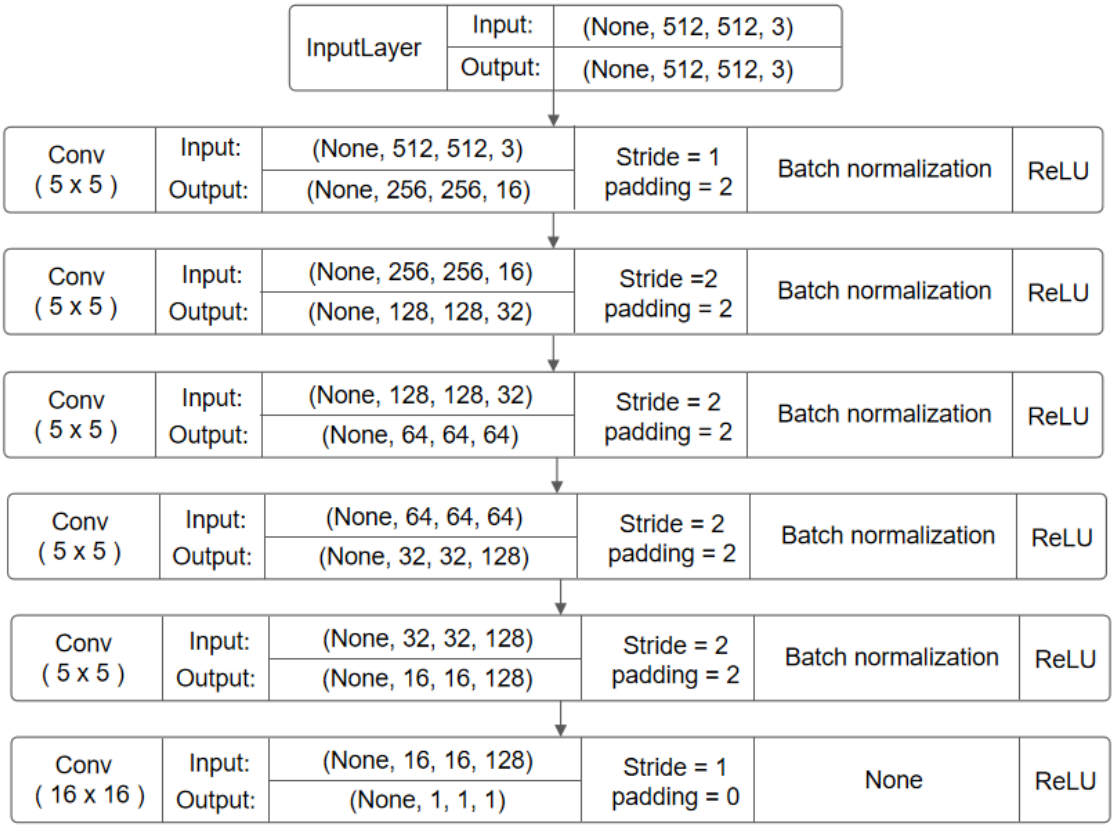


Figure 4.7 Network structure of our discriminator.

parallelly with the generator to distinguish between real items and counterfeits [91]. We utilize the bilateral grid network as our generator, and we build the discriminator with the network structure as illustrated in Figure 4.7. We used large receptive fields and strides in the network to extract global features better. Thus, the discriminator will not be restricted by local details.

The basic GAN is notoriously difficult to converge during training. Therefore, we utilized the WGAN-GP [63] for our training, which can efficiently prevent the model collapse and the vanishing gradient problem.

During training, the discriminator is supposed to learn from a high-quality collection of retouched photographs and distinguish the main properties contributing to a visually pleasing photograph. If the bilateral grid network keeps generating greyish outputs because of the

average effect, this characteristic is supposed to be identified by the discriminator. Therefore, the feedback from the discriminator can navigate the generator away from the 'preserved' retouching strategy. The overall information flow between the bilateral grid network and the discriminator is illustrated in Figure 4.8

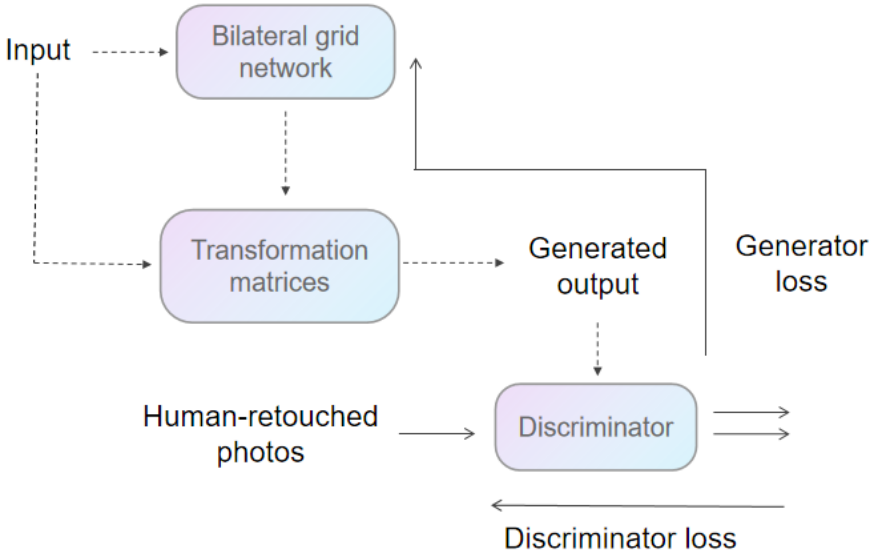


Figure 4.8 Overall information flow of the bilateral grid network and the discriminator.

4.3 Loss Function

4.3.1 Mean Squared Loss

We evaluate the quality of the generated images by comparing the difference between the human-retouched version and our generated output. This loss function facilitates the network to reproduce the retouching style demonstrated by the retouching operator. Let I_i be the original image, and I_g be the generated output from the bilateral grid network. I_i and I_g are both normalized to range [0,1]. The L_{MSE} is defined as:

$$L_{MSE} = ||I_i - I_g||^2 \tag{4.2}$$

4.3.2 Discriminator Loss

The discriminator loss provides a binary evaluation standard for the generator. The generator aims at getting a high score by cheating the discriminator, and it receives a large penalty if the discriminator can easily classify the generated images as counterfeits. Let G represent the bilateral grid generator, and D represent the discriminator. Assume P_i represents the batch of input images, the discriminator loss to update discriminator parameters θ is defined as:

$$L_d = \mathbb{E}_{I \sim P_i}[D(G(I)] \tag{4.3}$$

For the discriminator network D, the loss function is defined as the sum of discriminator loss and the gradient penalty. The gradient penalty helps to maintain the Lipschitz constraint of the discriminator. The discriminator receives generated outputs $\tilde{x}$ in batch P_g and expert-retouched photographs x in batch P_r. It is trained to give a high score for human-retouched images and a low score for generated images. The loss function is defined as:

$$\begin{aligned} L = & \mathbb{E}_{\tilde{x} \sim P_g}[D(\tilde{x})] - \mathbb{E}_{x \sim P_r}[D(x)] \\ & + \lambda \, \mathbb{E}_{\tilde{x} \sim P_{\hat{\infty}}}[(|| \bigtriangledown_{\hat{x}} D(\hat{x})||_2 - 1)^2] \end{aligned} \tag{4.4}$$

where we set the λ to 10 as suggested in [63], and $\hat{x}$ is obtained by sampling uniformly along the straight line between $\tilde{x}$ and x:

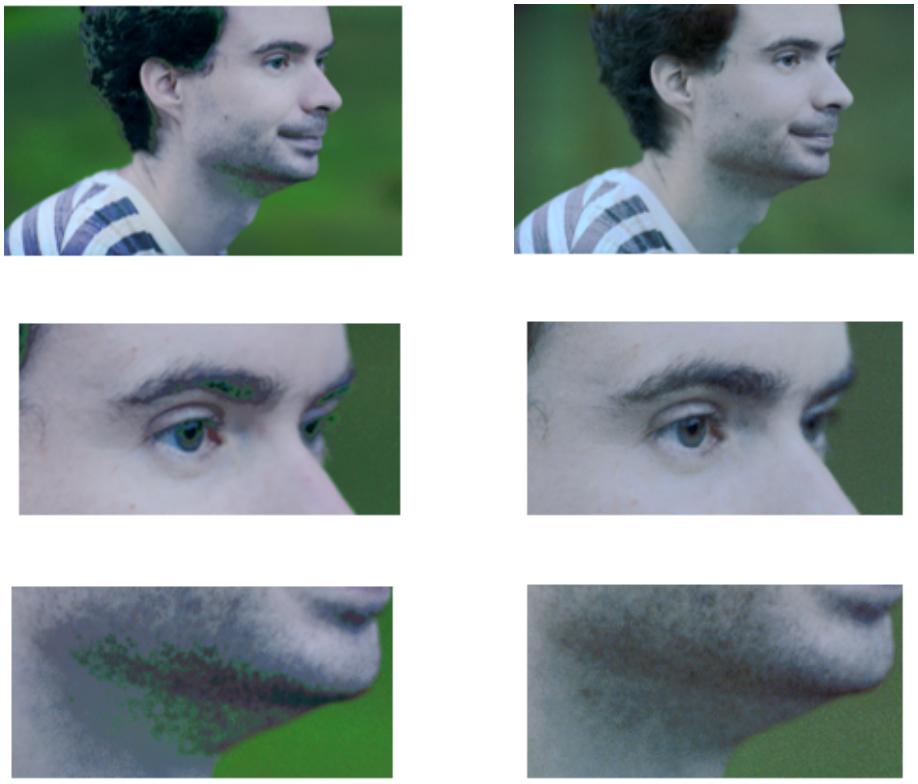
$$\hat{x} = t\tilde{x} + (1-t)x \tag{4.5}$$

where t is randomly selected in range $(0, 1)$.

4.3.3 Bilateral Loss

Our experimental results indicated that the introduction of a discriminator can result in vivid and saturated colour but sometimes bring unrealistic distortion, which seriously affects

the visual quality of photographs. To figure out the cause of distortion, we tested several grid sizes of the bilateral grid network from $8 \times 8 \times 8$ to $36 \times 36 \times 24$. During the test, other parameters like training epochs and learning rate are kept the same. The comparison between the output of different grid size is illustrated in Figures 4.9 and 4.10. Serious colour quantization and false contouring can be noticed in the output generated through the smaller grid, especially in hair areas like mustache and eyebrows.



(a) Grid with size $8 \times 8 \times 8$. **(b)** Grid with size $36 \times 36 \times 24$.

Figure 4.9 Comparison of the generated output Man between two different bilateral grid sizes.

The colour quantization is caused by the conflict between the discriminator and trilinear interpolation operation. To improve the dynamic range of input images, the feedback from the discriminator facilitates a large gap between adjacent transformation matrices in the bilateral grid. The large difference helps to enhance image contrast but sometimes transfers the continuous gradation to abrupt changes from one tone to another in generated images.

(a) Grid with size $8 \times 8 \times 8$. **(b)** Grid with size $36 \times 36 \times 24$.

Figure 4.10 Comparison of the generated output Dog between two different bilateral grid sizes.

Sometimes a small region contains multiple objects with distinct colours, but the pixels show similar intensity in an underexposed photograph. When the bilateral grid fails to provide a sufficient number of transformation matrices, the colorization of one object gets influenced by neighbouring elements.

A large bilateral grid with abundant affine transformation matrices can reduce the unintended artifact of colour quantization. However, a large bilateral grid requires high time and memory cost.

It is necessary to place additional restrictions on the difference between neighbouring affine transformation matrix stored in the bilateral grid to prevent posterization. Chen *et al.* [21] applies Gaussian filters on the spatial and range dimensions of the bilateral grid to maintain the smooth distribution of value maps. In our case, a network receives feedback from the loss function. Hence, instead of directly smoothing the generated bilateral grid, we propose a bilateral loss function to guarantee the piecewise smoothness on the none-edge regions. We calculate the derivatives on both horizontal and vertical directions of generated

image I_g and expert-retouched images I_t. When there is no edge in I_t, but a large gradient is detected in the same position in I_g, the network would receive a large penalty for introducing extra edges. The bilateral loss function would sum up the squared difference between the gradient images, and the formula is defined as:

$$\begin{aligned} L_B =& 1 - \sum_x \sum_y (f(\partial_x I_t)(\partial_x I_t - \partial_x I_g)^2 \\ &+ f(\partial_x I_t)(\partial_y I_t - \partial_y I_g)^2)/2, \end{aligned} \tag{4.6}$$

where $f(x)$ represents: $f(x) = 1 - \frac{1}{1+e^x}$

4.3.4 Imitation-to-innovation Training Scheme

To further reduce the conflict between the GAN and the bilateral grid, we introduce a new training strategy to mimic the learning process of human learners. As mentioned earlier, GAN is notoriously difficult to train because of the imbalance between the generator and the discriminator [63]. Non-convergence, vanishing gradient, and mode collapse are common obstacles in the GAN training process. WGAN-GP replaces the last sigmoid layer with a linear activation to facilitate the calculation for Wasserstein distance. However, a linear activation sometimes results in large fluctuations. Even with the gradient penalty, the discriminator loss will range from about -10 to 10, whereas the MSE loss usually ranges from 1×10^{-3} to 1×10^{-4}. From our experimental results, it is hard to determine a proper weight for the discriminator loss. The network failed to converge when we assigned a large weight to discriminator loss. Nevertheless, if a small weight was assigned to the discriminator loss, the discriminator does not make a large difference. Some failure cases are illustrated in Figure 4.11. Some unnatural colour blocks can be noticed in the output images.

During the experiment, we found that some manual modifications on the weight assigned to the discriminator loss during training can improve the quality of outputs. The modifications are similar to the human learning process. When learning image retouching, novice

retouchers usually begin by following tutorials and mimicking the retouching workflow of experts. Although there is no determined workflow for image retouching, novice retouchers can learn some general techniques, such as improving the colour contrast. Retouchers with some experience will move beyond simple mimics and step into spontaneous innovation regarding their aesthetic understanding. They usually receive feedback from their teachers or viewers on online platforms. The positive or negative comments from outside motivate the retouchers to adjust their strategy.

Figure 4.11 Outputs of the bilateral grid network when weight assigned to the discriminator loss is set as 0.1.

Similar to the learning process of human beginners, our network is required to duplicate the retouching strategy demonstrated by hired experts at the start of training. The weight assigned to the discriminator loss grows when the training proceeds. We use the learning rate decay as the training strategy, so the learning rate decreases while the weight of the discriminator loss increases. Assuming e_i denotes the number of training epochs, the weight assigned to discriminator loss w_D is calculated as follows:

$$w_D = \left(\frac{e_i^2}{1+e_i}\right) \times 10^{-3} \tag{4.7}$$

The overall loss function L is defined as the weighted sum of MSE loss L_M, discriminator loss L_D and bilateral loss L_B:

$$L = L_M + w_D L_D + L_B \tag{4.8}$$

By introducing the imitation-to-innovation training scheme and bilateral loss, the posterization artifacts in local areas are reduced. The colors of neighbouring elements have less

influence on each other. Thus, the system is capable of recovering plausible colorization in underexposed regions. Some comparisons will be given in Chapter 5.

Chapter 5

Experimental Results and Evaluation

5.1 Environment Setup

Python is a widely-used language in deep learning areas. Two popular deep learning frameworks, Tensorflow and Pytorch, have been developed on Python. Tensorflow has a wide application in industrial fields due to its earlier release. Nevertheless, its complex graph construction process makes it challenging to build and debug. Although TensorFlow 2 enables a high-level API extension called Keras, which provides simple interfaces for network flow design, it has some incompatibility with some older features. Although developed later, Pytorch provides convenient syntax and concise operations after several updates.

Due to the convenience of Pytorch, We built our network via Pytorch on the Windows system. We trained our network for 60 epochs with a mini-batch size, which equals to 4. Our training was carried out on an Intel-based PC equipped with one NVIDIA Geforce GTX 1080Ti. We used the bilateral grid data structure in our network design, so training a CNN on megapixel images only cost 4 to 5 days.

5.2 Experiment Settings

5.2.1 Datasets

Training Datasets:

Manual image enhancement requires operators to have professional skills. It is a heavy workload even for experts to edit a large bunch of photographs. Although lots of visually pleasing photographs can be found on photograph sharing platforms, photographers usually do not share the raw editions before retouching. Therefore, collecting a paired dataset for supervised training is expensive. Some methods rely on synthetic data for training. For instance, Park *et al.* [87] randomly applied distortion filters on high-quality photographs to obtained the paired raw photographs. There are only a limited number of public image enhancement datasets available at present, which are introduced below:

- MIT-Adobe FiveK Dataset: MIT-Adobe 5K was released by Adobe in [14]. The dataset contains 5000 photographs captured straight from cameras. The raw photographs mainly suffer from underexposure and unsaturated colour. This dataset consists of a broad diversity of contexts and environments. Five photography students (also referred to as retouching experts in this thesis) in an art school were hired to retouch each photo according to their own preference. MIT-Adobe 5K is the most popular dataset for image enhancement tasks.

- DPED dataset: Ignatov *et al.* [3] released a dataset containing a paired photograph collection with low-quality and corresponding high-quality photographs. The low-quality photographs were collected via built-in cameras in smartphones, such as BlackBerry or iPhone. The high-quality set was captured through Canon 70D. Due to physical limitations, the captured photographs are not perfectly aligned.

- See-in-the-Dark (SID) dataset: SID dataset is provided by Chen *et al.* in [4] for short-exposure images enhancement. It contains short-exposure images and a corresponding

long-exposure reference image. Paired images were captured by the same device fixed at one location with different exposure settings. The slight time gap causes unaligned content.

We utilized the raw images in the MIT-5K dataset as the inputs to our network. The MIT-5K dataset provides images in Adobe digital negative (DNG) format with 16-bit information. DNG is Adobe's proprietary image standard for raw images captured directly from camera sensors without processing. This format cannot be read by most third-party software or image processing libraries, such as Pillow. To process the image in Pytorch, we utilized Adobe Lightroom to convert the DNG files to TIFF format in RGB colour space with 8-bit colour depth.

To accelerate the training, we resized input images to 600×600 through bicubic interpolation, and the resized images were randomly cropped to 512×512 patches for training. The input to the bilateral grid generator was downsampled to 256×256. Our experimental results showed that images of size 256×256 were sufficient for the bilateral grid network to generate a retouching strategy and can reduce the time cost.

MIT-5K also contained five different retouched versions based on each raw image. The retouching is implemented by five hired experts A, B, C, D, and F. The user study in [14] indicated that expert C was most favoured. Therefore, we utilized the retouched version of expert C as the ground truth.

The MIT-5K dataset was collected in 2013, and most photographs in MIT-5K are retouched with a natural style to improve the colour saturation. However, some different retouching styles can be noticed in several online photograph sharing platforms. Instead of reproducing the visual scenes, some photographers retouch their photos to idealize their work and exaggerate some features. Some photographers demonstrated their favour for exaggerated contrast and vibrant colorization. Meanwhile, some people pursue subdued or even monochromatic photographs. These works are usually visually attractive but sometimes

render images less realistic.

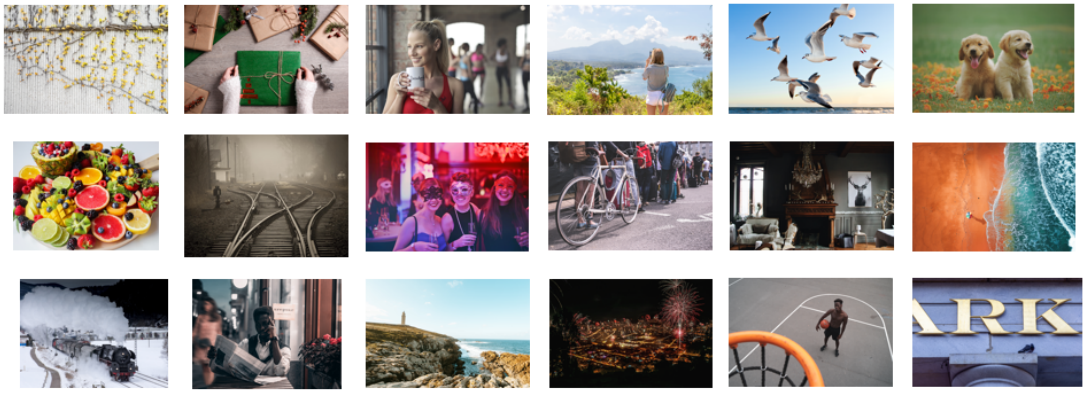


Figure 5.1 Some example from Flickr dataset.

Consequently, the MIT-5K dataset fails to reflect the current preference of retouching style. Instead of utilizing only the retouched photographs in MIT-5K as high-quality collections as did in [89], we created another dataset. We crawled 1000 high-rating photographs from the favourite list on Flickr, a free platform for photo sharing. We ensured the dataset containing a broad range of subjects, including natural scenery, portraits, and animals. We also kept the dataset containing various colorization and retouching styles. Some examples are illustrated in Figure 5.1

To maintain a balance between the natural editing style and the exaggerated style, we randomly picked up one example from the Flickr or MIT-5k retouched dataset and give the image to the discriminator. The image is resized to size 600×600 and then randomly cropped to size 512×512.

Test Datasets:

We separated the MIT-5k into two subsets, one contained 3000 images for training, and the other contained 2000 images for testing. During training, the test dataset was hidden from our network. To maintain the intact content of test images for visual quality evaluation, we kept their original size during testing.

5.2.2 Parameter Setting

We conducted a series of experiments to determine the optimal set of all coefficients and parameters. We utilized the Adam optimizer in the generator with $\beta = 0.999$, and RMSprop with $\beta = 0.999$ for the discriminator as suggested in [62]. The generator starts with learning rate $r = 2 \times 10^{-4}$, while the discriminator begins with $r = 2 \times 10^{-5}$. Both learning rates decay to 1×10^{-6} at the end.

We utilize the default scheme of Pytorch for parameter initialization, where Xavier is applied to the layer containing Tanh optimizer, and He is applied to layers with ReLU. As ReLU and LeakyReLU are the only two activation functions we used, the He initialization is applied to all layers.

5.3 Evaluation Metrics

To better evaluate the performance of our system, We implemented three state-of-the-art image enhancement methods for comparison. The three methods are based on different theories, and they demonstrated the best performance regarding the corresponding theories for both visual quality and time cost.

- JP [20]: JP was based on the Retinex theory, which demonstrates excellent performance in recovering information in underexposed regions. It iteratively decomposes illumination and reflection from input images.

- BIL [17]: The BIL network was trained via a supervised learning strategy. The network was trained to reproduce the retouching strategy of Expert C of the MIT-5K dataset. It can process images in real-time. This method demonstrated the most stable performance in photograph enhancement tasks.

- Whitebox [8]: Hu *et al.* utilized GAN for a semi-supervised learning scheme. The network was trained to apply predefined filters to an input image to improve its visual

Table 5.1 Quantitative comparison between different methods.

Method	PSNR(dB)	SSIM	Structural difference
JP [20]	24.96	0.792	0.937
BIL [17]	28.74	0.866	0.962
White-box [8]	24.42	0.822	0.943
Our Work	27.82	0.857	0.965

quality. Although the network theoretically can generate multiple solutions for one image, only one or two solutions provide enough perceptible details in images. We used the outputs with the clearest details for comparison.

BIL [17] and Whitebox [8] both utilized MIT-5K dataset as the train and test dataset. In addition, Hu *et al.* [8] crawled 766 photographs from two photographers on 500px.com and added them to retouched photographs in MIT-5k dataset to train the discriminator. This dataset has not been released due to the copyright issue, so we use the high-quality image crawled from Flickr as an alternative.

Aesthetic is a subjective topic. It is challenging to evaluate the attractiveness of one image quantitatively. Most evaluating metrics are reference-based, and they measure the perceived quality of generated images based on a reference image, such as the PSNR and the SSIM. We employed the PSNR and the SSIM to evaluate the performance of our network regarding the expert-retouched images and generated images. We compared our results with the above three enhancement methods. The results are shown in Table 5.1.

As BIL is trained to reproduce the reference photographs, it demonstrated the best results. The superiority is especially evident when tested with PSNR, which estimates absolute errors between pixels. SSIM measures the mean luminance, contrast and structural difference between reference images and test images. Because JP, Whitebox, and our network do not aim to reproduce the retouching style of a single human retoucher, the luminance and

Table 5.2 Comparison in terms of NIQE between different methods.

Method	NIQE
Original	3.9012
JP [20]	3.3377
BIL [17]	3.4196
White-box [8]	3.3597
Our Work	3.3471
Expert-retouched	3.3395

contrast of generated outputs usually have differences with the human retouched versions. As all four methods should maintain the same structure with the reference images, we also separately calculated the structural difference using Equ. 2.32. It can be noticed that our system demonstrated the best performance to maintain the original structural information.

Due to the multi-modality nature of photograph retouching, reference-based metrics are not fair for unsupervised learning models and non-learning-based algorithms. There are some no-reference based models for image quality assessment. One widely-used model is called Natural Image Quality Evaluator (NIQE) [16]. It is a blind perceptual model that can evaluate the image quality without prior knowledge about the reference image and possible distortions. It generates a score by comparing the distance of the multivariate Gaussian of natural scene statistics between a test image set and a high-quality image collection. We randomly selected 200 images from the expert-retouched test dataset and 100 images from the Flickr dataset to train the NIQE model. We tested the remaining 1800 images in the test dataset on NIQE, and the results are illustrated in Table 5.2. A low NIQE score represents good performance.

In Table 5.2, it can be noticed that JP demonstrates the best performance, even better than human retouched photographs. However, from the visual comparison, the outputs

of JP are usually subdued and unsaturated, and they are not superior compared to the manually retouched work. As indicated in [16], NIQE is sensitive to computer graphic editing and sometimes regards retouching as distortion. Consequently, it cannot fully provide an indicative evaluation for retouching quality.

To date, there is still no reliable objective evaluation system to judge the visual quality. Thus, to compare the performance of different systems, we conducted a user study and asked humans to get involved in image quality evaluation as did in [8, 85].

5.3.1 User Study

To obtain a reliable evaluation of the performance of our network, we conducted a user study on Amazon Mechanical Turk (AMT) [92]. AMT is a crowdsourcing marketplace for subjective research, such as market surveys or data annotations. Tasks are completed by distributed workers over the Internet, and workers receive a small reward for every action they perform. We have obtained the ethics approval from the Office of Research Ethics and Integrity at University of Ottawa. The ethics file number is H-12-20-6411.

The workers on AMT are registered users who are required to be at least 18-years-old. The workers are people who want to make money in their spare time. In AMT, the requesters who publish tasks have the opportunity to evaluate the completed work and decide whether to approve it or decline it. The approval rate is recorded for each worker. When publishing a task, the requesters can specify additional requirements. Workers must meet the requirements to work on these tasks. To ensure our user studies to be completed in high quality, we require the participants to have an approval rate of 90 percent or above. Because of the confidentiality term of AMT, the personal information of participants is hidden from the requesters.

We randomly selected 112 raw images from our test dataset, and we implemented our method as well as the other three methods on this collection to generate 448 images in total.

The layout of the web page for the study was designed, as shown in Figure 5.2. During the user study, one image is displayed to the participants each time. Participants were asked to rate the quality of the generated images according to the given questions. There are four questions: (Q1) 'are the details in the image perceptible?'; (Q2) 'Is this image visually realistic?'; (Q3) 'Does this image suffer from underexposure or overexposure?'; and (Q4) 'Is this image visually pleasing?'. Participants were given five options from 1 to 5, where 5 represented perfect, and 1 represents bad quality.

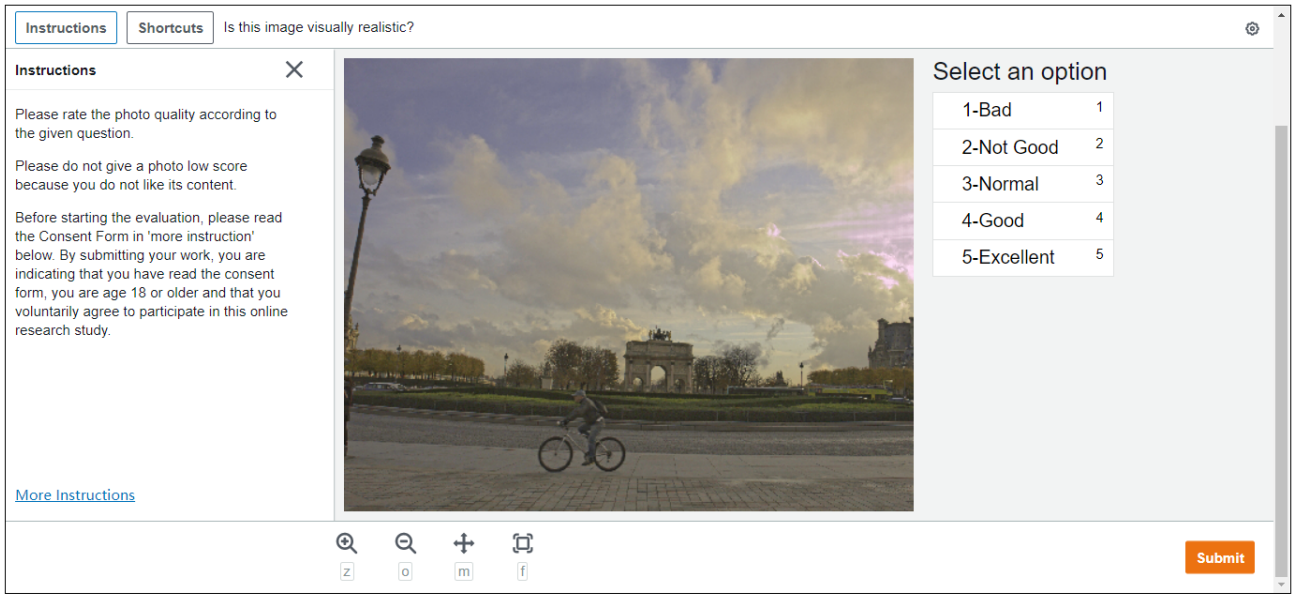
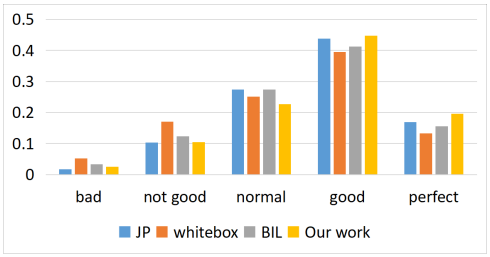
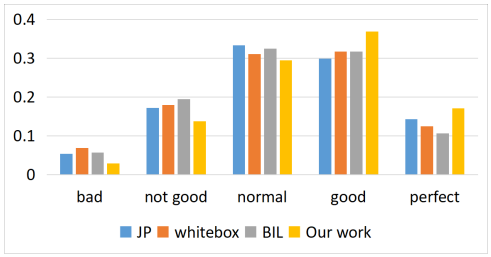


Figure 5.2 The layout of our user study page.

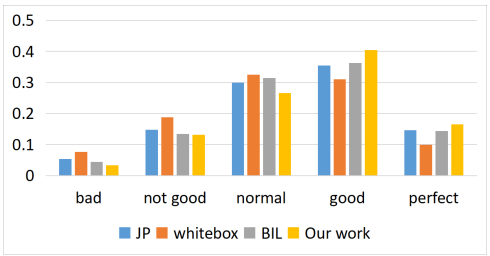
Because the AMT only supports one question for a task, we recruited 50 workers for each question. Instead of asking participants to evaluate 448 images at one time, we give each participant $\frac{1}{4}$ of images in case that the judgement of participants is influenced by fatigue. To exclude participants who made rash decisions to save time cost, we rejected several submissions that were completed within one minute. In addition, We shuffled the exhibition order of every four images with the same content in case a participant becomes sensitive to underlying retouched patterns in different methods.



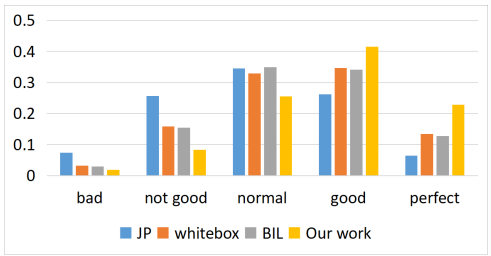
(a) 'are the details in the image perceptible?'



(b) 'Is this image visually realistic?'

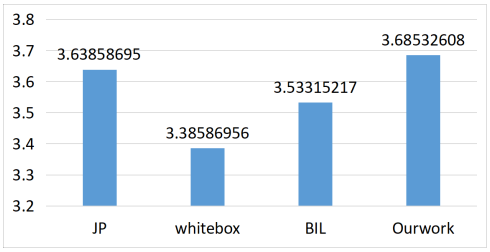


(c) 'Does this image suffer from underexposure or overexposure?'

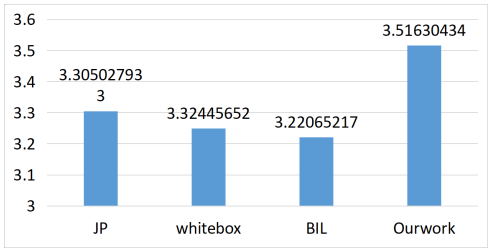


(d) 'Is this image visually pleasing?'

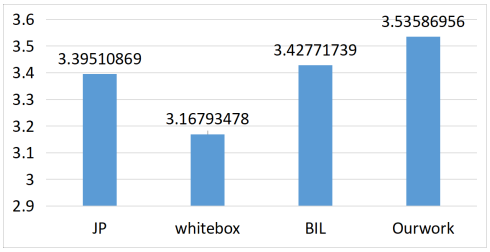
Figure 5.3 Rate distributions of our user study.



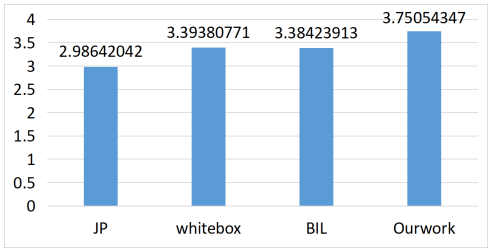
(a) 'are the details in the image perceptible?'



(b) 'Is this image visually realistic?'



(c) 'Does this image suffer from underexposure or overexposure?'



(d) 'Is this image visually pleasing?'

Figure 5.4 Average rating scores of our user study.

We gathered the scores to evaluate our performance compared with other methods. The distributions of the scores and average scores are illustrated in Figures 5.3 and 5.4. It can be noticed that our method obtained the highest rates from human participants compared with other methods on all features. Our system can generate visually pleasing and realistic results while maintaining perceptible details from underexposed areas. In addition, although JP obtains the best NIQE, it obtained the lowest score for 'visual pleasing' (Q4). The contrast reveals that NIQE score cannot fully reflect visual aesthetics preference. The design of an indicative evaluator for photo quality is still challenging today.

5.3.2 Time Cost Comparison

One of the major obstacles to photograph enhancement is the high time cost for processing multi-megapixel images. To demonstrate that our proposed method can efficiently reduce time costs, we compared the time cost of our methods to the other three approaches.

We randomly selected 100 images from MIT-5K dataset. To prevent the out-of-memory error because of the high memory cost of the Whitebox method [8], we randomly cropped the input image to size 1920×1080. As the time cost may fluctuate slightly for each image, we calculated the average time cost for every example, and the results are illustrated in Table 5.3. It can be noticed that only our method and BIL [17] successfully reduced the time cost to microseconds, while the other methods required several seconds to process the megapixel image on a GPU.

5.3.3 Visual Comparison

To compare the performance of different methods, we picked several raw images from MIT-5K dataset and implemented different methods on these datasets. The obtained outputs are illustrated in Figures 5.5, 5.6, 5.7, 5.8, and 5.9.

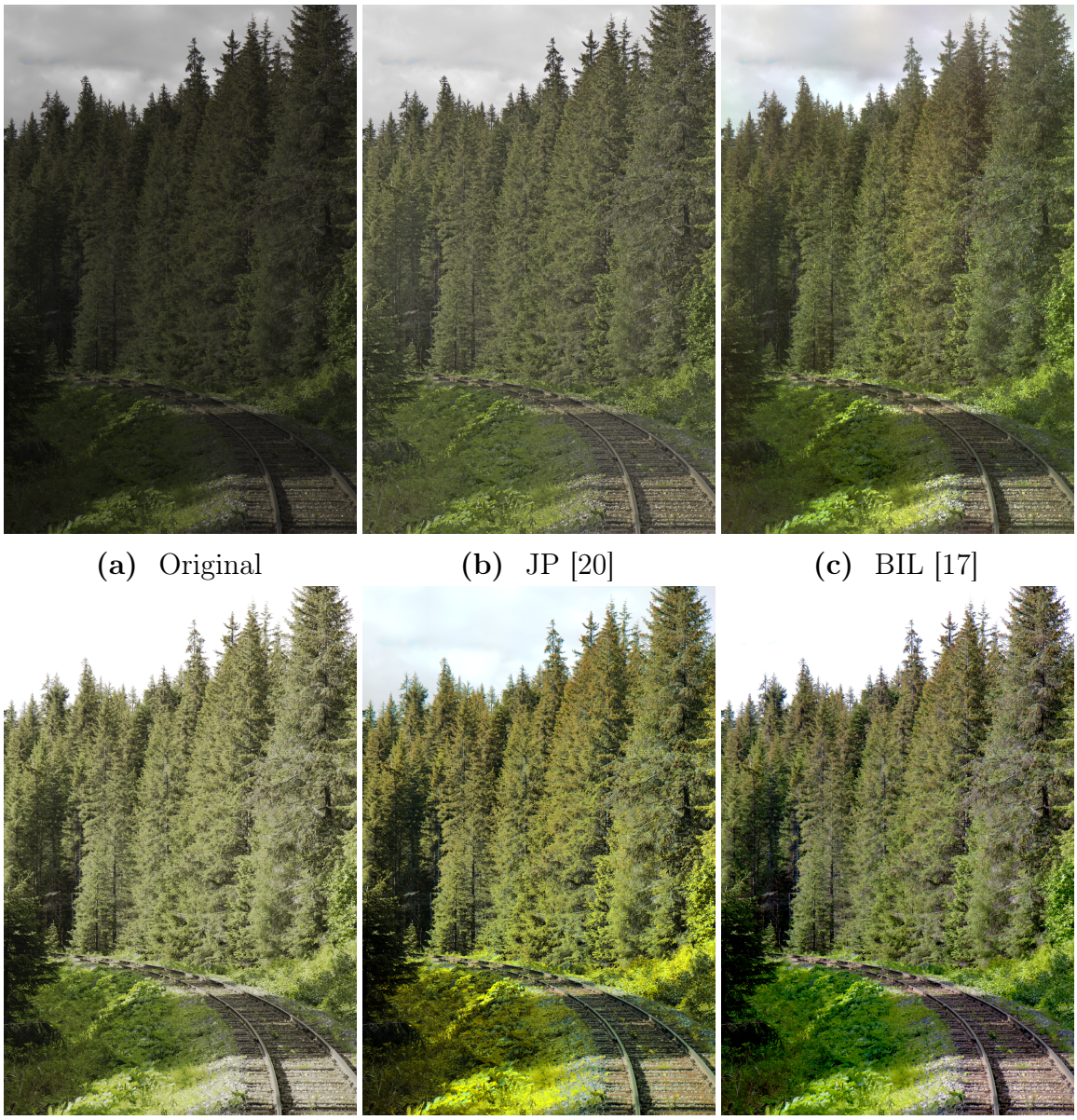
From the visual comparison, it can be noticed that our method generated more vibrant

Table 5.3 Comparison in terms of time cost between different methods.

Method	Average Time Cost
JP [20]	6.75s
BIL [17]	14.10ms
White-box [8]	5.54s
Our Work	16.52ms

and natural colorization compared with other works. Our system successfully combined the superiority of simplified one-solution training strategy and semi-unsupervised learning. It is capable of generating vibrant colours compared to desaturated colorizations caused by averaging effect, such as [17]. Moreover, our approach demonstrated a more stable performance than simply utilizing GAN for unsupervised learning. It is capable of recovering imperceptible details while maintaining the contrast of dark and light regions, while JP [20] will reduce the contrast to recover information in extreme dark parts, which may render the image less stereo as illustrated in Figure 5.9. It can also be noticed that our network is not restricted to a single tone. It can process diverse contents, such as oceans, forests, flowers, portraits, and night scenes.

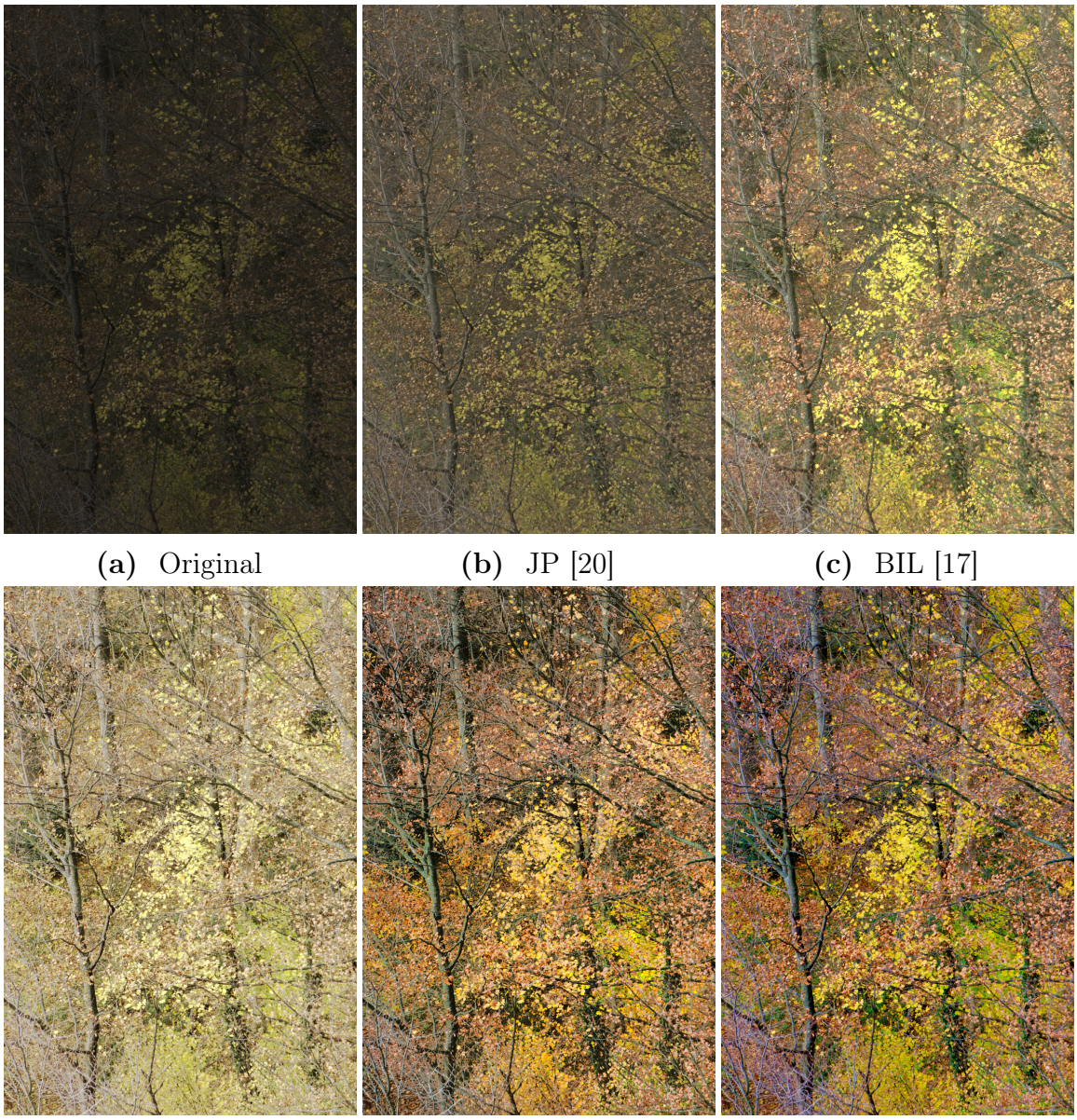
Besides, our system can produce compatible tones for realistic portraits. Portrait processing is always challenging among image enhancement tasks. Humans have more acute insight into human faces. A slight distortion, like shape contrast or unsuitable tone, usually result in an unrealistic feeling. Besides, the solution should handle the fine-grained and multifold details of every person like skin tone, freckles, hair, or the shadow cased by a dimple. Compared with other methods, our method is capable of generating natural portraits and recovering imperceptible details, as shown in Figure 5.7.



(a) Original **(b)** JP [20] **(c)** BIL [17]

(d) White-box [8] **(e)** Our work **(f)** Expert-retouched

Figure 5.5 Visual comparison between different methods on photograph Forest.



(a) Original **(b)** JP [20] **(c)** BIL [17]

(d) White-box [8] **(e)** Our work **(f)** Expert-retouched

Figure 5.6 Visual comparison between different methods on photograph Fall Forest.

(a) Original **(b)** JP [20] **(c)** BIL [17]

(d) White-box [8] **(e)** Our work **(f)** Expert-retouched

Figure 5.7 Visual comparison between different methods on photograph Father-and-son.

(a) Original

(b) JP [20]

(c) BIL [17]

(d) White-box [8]

(e) Our work

(f) Expert-retouched

Figure 5.8 Visual comparison between different methods on photograph Shopping.

(a) Original

(b) JP [20]

(c) BIL [17]

(d) White-box [8]

(e) Our work

(f) Expert-retouched

Figure 5.9 Visual comparison between different methods on photograph Street.

5.3.4 Comparison between with or without Bilateral Loss and Imitation-to-innovation Training Strategy

In the network training, we propose the bilateral loss and imitation-to-innovation training strategy to prevent noise enhancement, posterization artifacts and unnatural colour. To prove that our strategy can efficiently reduce the distortion, we trained one network with the same network structure but without the bilateral loss and imitation-to-innovation training strategy. The loss function is defined as the sum of discriminator loss L_D and MSE loss L_M as:

$$L = L_M + 0.03L_D \tag{5.1}$$

The two networks were trained for 60 epochs with a mini-batch size of 4, which is the same as our proposed network. The comparison is illustrated in Figures 5.10, 5.11, 5.12 and 5.13. It can be noticed that without bilateral loss and imitation-to-innovation training strategy, the system sometimes brings unintended artifact and generates unrealistic photographs. In Figures 5.10, our system reconstructs the continuous gradation of shadow on human face, while the system without our training scheme and loss function introduces abrupt change of tones. In Figure 5.11, our system smooths the yellow noise on the neck. The unnatural color on sky in Figure 5.12 and on hair in Figure 5.13 are also prevented by our training scheme and loss function. By comparison, it can be noticed that our method with bilateral loss and imitation-to-innovation training scheme can successfully prevent posterization and noise enhancement.

(a) Without **(b)** With

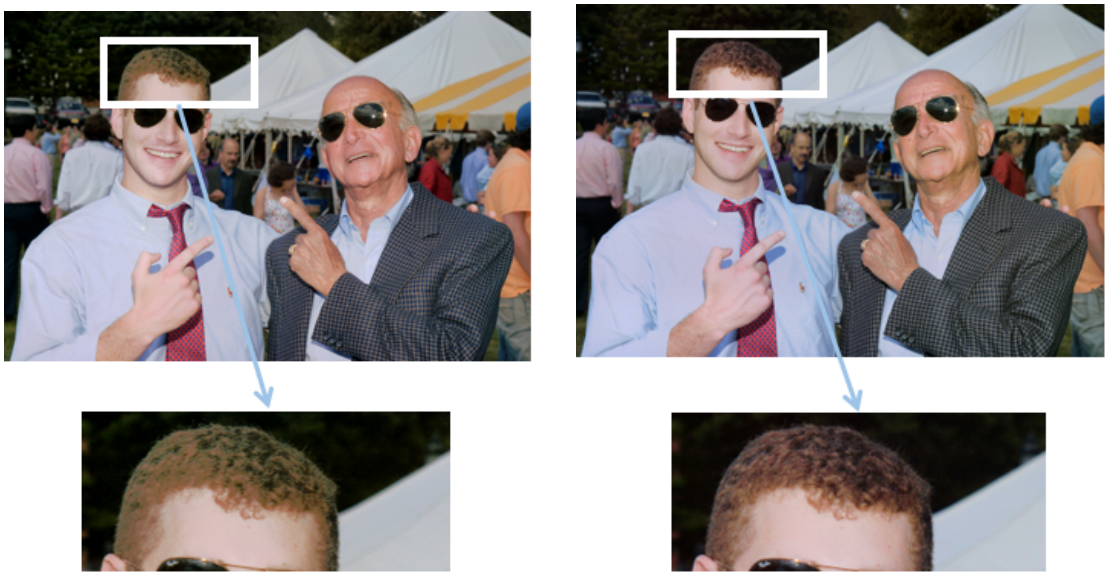
Figure 5.10 Visual comparison of face shadow between with or with using our proposed bilateral loss function and imitation-to-innovation training scheme.

(a) Without **(b)** With

Figure 5.11 Visual comparison of face details between with or with using our proposed bilateral loss function and imitation-to-innovation training scheme.

(a) Without **(b)** With

Figure 5.12 Visual comparison of sky color between with or with using our proposed bilateral loss function and imitation-to-innovation training scheme.



(a) Without **(b)** With

Figure 5.13 Visual comparison of hair color between with or with using our proposed bilateral loss function and imitation-to-innovation training scheme.

Chapter 6

Conclusion

In the thesis, we proposed a new real-time photograph enhancement system by combining the bilateral grid network with GAN. The MSE loss and discriminator function provide binary metrics to evaluate the quality of photographs. Our proposed method overcomes the one-to-one mapping restriction and succeeds in generating images that can satisfy global aesthetic value. Our contributions can be summarized as follows:

- We integrated the bilateral grid data structure and generative adversarial network in image transformation area for the first time. The discriminator can facilitate our bilateral grid generator to learn from an unpaired collection of high-quality images without introducing additional time cost.

- To maintain an effective combination of bilateral data structure and discriminator loss feedback, we replace the batch normalization layer with instance normalization to remove the mutual influence of different distribution. Besides, we introduced an additional bilateral loss function to force smooth affine matrix distribution in the vertical direction on the bilateral grid. Compared with simply using bilateral grid or using GAN to train the system, our network demonstrates high performance in photograph enhancement. It can recover vivid colors from underexposed photographs.

(a) Original **(b)** Expert-retouched **(c)** JP [20] **(d)** Our work

Figure 6.1 A failure case of our system on photograph Orchid.

There are still limitations for our system. When processing photo captured in extremely dark environments, our system sometimes generates plausible color which has a large difference with the real scenes. One example is illustrated in Figure 6.1. In this case, our system fails to display the original texture and colour of the orchids. Besides, the datasets for photograph enhancement are limited. The raw photographs in MIT-5K dataset mainly suffer from underexposure and subdued color. Thus, our network has limitations on fixing overexposed photographs. Afifi *et al.* [93] collected 24330 photographs under multiple exposure problems. However, this dataset has not been released to the public. After its release, we will train our network with this dataset and evaluate the performance of our system.

In the future, we would like to work on some practical applications of our system by deploying it on cloud and testing its performance as a mobile app. We would also work on the further enhancement of our network to handle images captured in extremely dark situations. Retinex theory demonstrates good performance in information recovery of underexposed images by removing the effect of illumination from reflection. The reflection information represents the physical characteristics of objects. We will try to first decompose reflection from images, and then train our network to process the reflection. Thus, the network may be able to better maintain the characteristics and prevent noise enhancement. One challenge for the implementation is the high time cost of Retinex algorithm. We will work on the integration of Retinex theory and neural networks while reducing the time complexity.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[2] Joon-Young Lee, Kalyan Sunkavalli, Zhe Lin, Xiaohui Shen, and In So Kweon. Automatic content-aware color and tone stylization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2470–2478, 2016.

[3] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3277–3285, 2017.

[4] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3291–3300, 2018.

[5] Yuen Peng Loh, Xuefeng Liang, and Chee Seng Chan. Low-light image enhancement using gaussian process for features retrieval. *Signal Processing: Image Communication, Special Issue on The Deep Learning in Computational Photography*, 74:175–190, May 2019.

[6] Yu-Sheng Chen, Yu-Ching Wang, Man-Hsin Kao, and Yung-Yu Chuang. Deep photo enhancer: Unpaired learning for image enhancement from photographs with gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6306–6314, 2018.

[7] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. *arXiv preprint arXiv:1906.06972*, 2019.

[8] Yuanming Hu, Hao He, Chenxi Xu, Baoyuan Wang, and Stephen Lin. Exposure: A white-box photo post-processing framework. *ACM Transactions on Graphics (TOG)*, 37(2):1–17, 2018.

[9] Huikai Wu, Shuai Zheng, Junge Zhang, and Kaiqi Huang. Fast end-to-end trainable guided filter. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1838–1847, 2018.

[10] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[11] Soonmin Bae, Sylvain Paris, and Frédo Durand. Two-scale tone management for photographic look. *ACM Transactions on Graphics (TOG)*, 25(3):637–645, 2006.

[12] Huanjing Yue, Jingyu Yang, Xiaoyan Sun, Feng Wu, and Chunping Hou. Contrast enhancement based on intrinsic image decomposition. *IEEE Transactions on Image Processing*, 26(8):3981–3994, 2017.

[13] Glenn D Hines, Zia-ur Rahman, Daniel J Jobson, and Glenn A Woodell. Dsp implementation of the retinex image enhancement algorithm. In *Visual Information Processing XIII*, volume 5438, pages 13–24. International Society for Optics and Photonics, 2004.

[14] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. Learning photographic global tonal adjustment with a database of input/output image pairs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 97–104. IEEE, 2011.

[15] Jianzhou Yan, Stephen Lin, Sing Bing Kang, and Xiaoou Tang. Change-based image cropping with exclusion and compositional features. *International Journal of Computer Vision*, 114(1):74–87, 2015.

[16] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012.

[17] Michaël Gharbi, Jiawen Chen, Jonathan T Barron, Samuel W Hasinoff, and Frédo Durand. Deep bilateral learning for real-time image enhancement. *ACM Transactions on Graphics (TOG)*, 36(4):1–12, 2017.

[18] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3985–3993, 2017.

[19] Qi Shan, Zhaorong Li, Jiaya Jia, and Chi-Keung Tang. Fast image/video upsampling. *ACM Transactions on Graphics (TOG)*, 27(5):1–7, 2008.

[20] Bolun Cai, Xianming Xu, Kailing Guo, Kui Jia, Bin Hu, and Dacheng Tao. A joint intrinsic-extrinsic prior model for retinex. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4000–4009, 2017.

[21] Jiawen Chen, Sylvain Paris, and Frédo Durand. Real-time edge-aware image processing with the bilateral grid. *ACM Transactions on Graphics (TOG)*, 26(3):103–es, 2007.

[22] Michael Elad. On the origin of the bilateral filter and ways to improve it. *IEEE Transactions on Image Processing*, 11(10):1141–1151, 2002.

[23] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998.

[24] Buyue Zhang and Jan P Allebach. Adaptive bilateral filter for sharpness enhancement and noise removal. *IEEE Transactions on Image Processing*, 17(5):664–678, 2008.

[25] BK Shreyamsha Kumar. Image fusion based on pixel significance using cross bilateral filter. *Signal, Image and Video Processing*, 9(5):1193–1204, 2015.

[26] Chunxia Xiao and Jiajia Gan. Fast image dehazing using guided joint bilateral filter. *The Visual Computer*, 28(6-8):713–721, 2012.

[27] Sylvain Paris and Frédo Durand. A fast approximation of the bilateral filter using a signal processing approach. In *European Conference on Computer Vision*, pages 568–580. Springer, 2006.

[28] Frédo Durand and Julie Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pages 257–266, 2002.

[29] Johannes Kopf, Michael F Cohen, Dani Lischinski, and Matt Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (ToG)*, 26(3):96–es, 2007.

[30] Jiawen Chen, Andrew Adams, Neal Wadhwa, and Samuel W Hasinoff. Bilateral guided upsampling. *ACM Transactions on Graphics (TOG)*, 35(6):1–8, 2016.

[31] Christian Richardt, Douglas Orr, Ian Davies, Antonio Criminisi, and Neil A Dodgson. Real-time spatiotemporal stereo matching using the dual-cross-bilateral grid. In *European Conference on Computer Vision*, pages 510–523. Springer, 2010.

[32] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[33] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[34] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on International Conference on Machine Learning*, pages 807–814, 2010.

[35] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[36] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[37] George E Dahl, Tara N Sainath, and Geoffrey E Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8609–8613. IEEE, 2013.

[38] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural Networks for Perception*, pages 65–93. Elsevier, 1992.

[39] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[40] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of International Conference on Computational StatisticsParis France*, pages 177–186. Springer, 2010.

[41] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[42] Andrew Cotter, Ohad Shamir, Nati Srebro, and Karthik Sridharan. Better mini-batch algorithms via accelerated gradient methods. In *Advances in Neural Information Processing Systems*, pages 1647–1655, 2011.

[43] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

[44] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[45] Yoshua Bengio and MONTREAL CA. Rmsprop and equilibrated adaptive learning rates for nonconvex optimization. *Computing Research Repository (corr abs)/1502.04390*, 2015.

[46] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[47] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[48] Ossama Abdel-Hamid, Li Deng, and Dong Yu. Exploring convolutional neural network structures and optimization techniques for speech recognition. In *Interspeech*, volume 11, pages 73–75, 2013.

[49] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[51] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[52] Siddharth Krishna Kumar. On weight initialization in deep neural networks. *arXiv preprint arXiv:1704.08863*, 2017.

[53] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[54] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[55] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

[56] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[57] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[58] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2089–2093. IEEE, 2017.

[59] Tingting Li, Ruihe Qian, Chao Dong, Si Liu, Qiong Yan, Wenwu Zhu, and Liang Lin. Beautygan: Instance-level facial makeup transfer with deep generative adversarial network. In *Proceedings of the 26th ACM International Conference on Multimedia*, pages 645–653, 2018.

[60] Yifan Liu, Zengchang Qin, Zhenbo Luo, and Hua Wang. Auto-painter: Cartoon image generation from sketch by using conditional generative adversarial networks. *arXiv preprint arXiv:1705.01908*, 2017.

[61] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[62] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[63] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[64] SS Vallender. Calculation of the wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications*, 18(4):784–786, 1974.

[65] Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability Theory and Related Fields*, 162(3-4):707–738, 2015.

[66] James Hegarty, John Brunhaver, Zachary DeVito, Jonathan Ragan-Kelley, Noy Cohen, Steven Bell, Artem Vasilyev, Mark Horowitz, and Pat Hanrahan. Darkroom: compiling high-level image processing code into hardware pipelines. *ACM Transactions on Graphics*, 33(4):144–1, 2014.

[67] Ravi Teja Mullapudi, Andrew Adams, Dillon Sharlet, Jonathan Ragan-Kelley, and Kayvon Fatahalian. Automatically scheduling halide image processing pipelines. *ACM Transactions on Graphics (TOG)*, 35(4):1–11, 2016.

[68] Erhan Alparslan and Mr Fuatince. Image enhancement by local histogram stretching. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(5):376–385, 1981.

[69] Yeong-Taeg Kim. Contrast enhancement using brightness preserving bi-histogram equalization. *IEEE transactions on Consumer Electronics*, 43(1):1–8, 1997.

[70] Stephen M Pizer, E Philip Amburn, John D Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart ter Haar Romeny, John B Zimmerman, and Karel Zuiderveld. Adaptive histogram equalization and its variations. *Computer Vision, Graphics, and Image Processing*, 39(3):355–368, 1987.

[71] Jianbing Shen, Xiaoshan Yang, Yunde Jia, and Xuelong Li. Intrinsic images using optimization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3481–3487. IEEE, 2011.

[72] Edwin H Land and John J McCann. Lightness and retinex theory. *Josa*, 61(1):1–11, 1971.

[73] Daniel J Jobson, Zia-ur Rahman, and Glenn A Woodell. Properties and performance of a center/surround retinex. *IEEE Transactions on Image Processing*, 6(3):451–462, 1997.

[74] Daniel J Jobson, Zia-ur Rahman, and Glenn A Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image Processing*, 6(7):965–976, 1997.

[75] Zia-ur Rahman, Daniel J Jobson, and Glenn A Woodell. Retinex processing for automatic image enhancement. *Journal of Electronic Imaging*, 13(1):100–111, 2004.

[76] Kobus Barnard and Brian Funt. Investigations into multi-scale retinex. 1998.

[77] Michael K Ng and Wei Wang. A total variation model for retinex. *SIAM Journal on Imaging Sciences*, 4(1):345–365, 2011.

[78] Berthold KP Horn. Determining lightness from an image. *Computer Graphics and Image Processing*, 3(4):277–299, 1974.

[79] Xueyang Fu, Delu Zeng, Yue Huang, Xiao-Ping Zhang, and Xinghao Ding. A weighted variational model for simultaneous reflectance and illumination estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2782–2790, 2016.

[80] Sing Bing Kang, Ashish Kapoor, and Dani Lischinski. Personalization of image enhancement. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1799–1806. IEEE, 2010.

[81] YiChang Shih, Sylvain Paris, Connelly Barnes, William T Freeman, and Frédo Durand. Style transfer for headshot portraits. 2014.

[82] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.

[83] Zhen Li, Jinglei Yang, Zheng Liu, Xiaomin Yang, Gwanggil Jeon, and Wei Wu. Feedback network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3867–3876, 2019.

[84] Yanyun Qu, Yizi Chen, Jingying Huang, and Yuan Xie. Enhanced pix2pix dehazing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8160–8168, 2019.

[85] Ruixing Wang, Qing Zhang, Chi-Wing Fu, Xiaoyong Shen, Wei-Shi Zheng, and Jiaya Jia. Underexposed photo enhancement using deep illumination estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6849–6857, 2019.

[86] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[87] Jongchan Park, Joon-Young Lee, Donggeun Yoo, and In So Kweon. Distort-and-recover: Color enhancement using deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5928–5936, 2018.

[88] Viacheslav Voronin, Evgenii Semenishchev, Vladimir Frants, and Sos Agaian. Smart cloud system for forensic thermal image enhancement using local and global logarithmic transform histogram matching. In *2018 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 153–157. IEEE, 2018.

[89] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134, 2017.

[90] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi 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)*, 35(4):1–11, 2016.

[91] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[92] Amazon mechanical turk. https://www.mturk.com/. Last accessed: 2020-12-29.

[93] Mahmoud Afifi, Konstantinos G Derpanis, Björn Ommer, and Michael S Brown. Learning multi-scale photo exposure correction. *arXiv preprint arXiv:2003.11596*, 2020.