Deep Learning-Inspired Image Quality Enhancement

Ruxin Wang

Faculty of Engineering and Information Technology

University of Technology Sydney

A thesis submitted for the degree of

Doctor of Philosophy

2017

To my loving parents
Jianxia Liu and *Jigang Wang*

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Ruxin Wang

Acknowledgements

I would like to sincerely thank everyone who has helped me to finish my doctoral studies.

First of all, I would like to express my sincere appreciation and deep gratitude to my supervisor **Prof. Dacheng Tao**. He has given me trust and freedom to pursue my research interests, and provided constructive suggestions to help me out of difficulties. I can always benefit and learn a lot from various detailed discussions with him, and be excited and energised by his amazing insight, unlimited patience, generous support, and constant encouragement. I feel very lucky to have had him as my supervisor.

I also wish to express my sincere appreciation to **Prof. Xinge You** who was my advisor when I was in Huazhong University of Science and Technology for master study. I am so grateful for his guidance and support. I would have not come to the research field of image processing and computer vision, and would not have met Prof. Dacheng Tao and come to UTS without him.

I would also like to give special thanks to my excellent collaborators: Prof. Jun Yu, Prof. Richang Hong, Dr. Kun Zeng, and Dr. Xiaoyan Li for their brilliant work and timely support. I have also been fortunate to work and have discussions with many other brilliant researchers: Dr Chang Xu, Dr Nannan Wang, Dr Fei Gao, Assistant Professor Chaohui Wang, Prof. Chen Gong, Dr Yong Luo, Dr Tao Liu, Prof. Kaibing Zhang, Dr Lianyang Ma, A/Prof Chenping Hou, Dr Xiao Liu, Dr Weilong Hou, A/Prof Bo Du, Prof Shengzheng Wang, A/Prof Tao Lei, A/Prof Wankou Yang, A/Prof Shigang Liu, A/Prof Xianye Ben, A/Prof Xiong Wang, and Dr Qiong Wang. I wish to

express appreciation to all of them for their support and kind companion.

I am so grateful to my colleagues and friends I met in Sydney: Mingming Gong, Tongliang Liu, Shaoli huang, Qiang Li, Zhibin Hong, Meng Fang, Changxing Ding, Maoying Qiao, Zhe Xu, Long Lan, Wei Bian, Tianyi Zhou, Jun Li, Chunyang Liu, Bozhong Liu, Zhiguo Long, Guodong Long, Jing Jiang, Huan Fu, Baosheng Yu, Zhe Chen, Zijing Chen, Xiyu Yu, Liu Liu, Yali Du, Guoliang Kang, Hao Xiong, Jiayan Qiu, Jiankang Deng, Chaoyue Wang, Dayong Tian, Jiang Bian, Bo Han, Kede Ma, Tianrong Rao, Lingxiang Wu, Sujuan Hou, Xiaoqing Yin, Peicheng Zhou, Mingjin Zhang, Tao Zhang, Haifeng Liu, Peng Hao, Haishuai Wang, and Xun Yang. All my friends here have provided strong support during both happy and stressful time. I am also incredibly grateful to my old friends: Chengbin Yan, Digang Wang, Xinwei Liu, Zhe Wang, Hao Ding, Junge Shen, Xin Wang, Ting Yuan, Haoran Lv, and Yuannan Zhao, who have accompanied me from my bachelor's degree to doctoral studies. I owe my deepest thanks to all of them!

Finally, I would like to express deep-felt gratitude to my family: my parents, my grandparents, my uncles and aunties, and my cousins, for their endless love, trust, encouragement, and full support throughout my studies and life.

I dedicate this thesis to them.

Abstract

Enhancing image quality is a classical image processing problem that has received plenty of attention over the past several decades. A high-quality image is always expected in various vision tasks, and degradations such as noise, low-resolution, and blur are required to be removed. While the conventional techniques for this task have achieved great progress, the recent top performer, deep models, can substantially and significantly boost performance compared with conventional ones. The advantages of deep learning which enables it to achieve such success are its high representational capacity and the strong nonlinearity of the models. In this thesis, we explore the development of advanced deep models for image quality enhancement by researching several fundamental issues with different motivations.

In particular, we are first motivated by a pivotal property of the human perceptual system that similar visual cues can stimulate the same neuron to induce similar neurological signals. However, image degradations can result in the fact that similar local structures in images exhibiting dissimilar observations. While the conventional neural networks do not consider this important property, we develop the (stacked) non-local auto-encoder which exploits self-similar information in natural images for enhancing the stability of signal propagation in the network. It is expected that similar structures should induce similar network propagation. This is achieved by constraining the difference between the hidden representations of non-local similar image blocks during training. By applying the proposed model to image restoration, we then develop a "collaborative stabilisation" step to further rectify forward propagation.

When applying deep models to image quality enhancement tasks, we are concerned about which factor, receptive field size or model depth, is more critical. To determine the answer, we focus on the single image super-resolution task, and propose a strategy based on dilated convolution to investigate how the two factors affect the performance. Our findings from exhaustive investigations suggest that single image super-resolution is more sensitive to the changes of receptive field size than to model depth variations, and that the model depth must be congruent with the receptive field size to produce improved performance. These findings inspire us to design a shallower architecture which can save computational and memory cost while preserving comparable effectiveness with respect to a much deeper architecture.

Finally, we study the general non-blind image deconvolution problem. It is observed in practice that by using existing deconvolution techniques, the residual between the sharp image and the estimation is highly dependent on both the sharp image and the noise. These techniques require the construction of different restoration models for different blur kernels and noises, inducing low computational efficiency or highly redundant model parameters. Thus, for general purposes, we propose a method by designing a very deep convolutional neural network which can handle different kernels and noises, while preserving high effectiveness and efficiency. Instead of directly outputting the deconvolved results, the model predicts the residual between a pre-deconvolved image and the corresponding sharp image, which can make the training easier and obtain restored images with suppressed artifacts.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Modern science and techniques equip us with the ability to capture images in our daily life. It is no exaggeration to say that images are ubiquitous and indispensable to us since they record and store snapshots of the world, just as the human visual system perceives the world. The applications related to images are innumerable, such as artistic photography, computer vision, remote sensing, astronomy, medical imaging, and microscopy. In each case, the obtained images produce a source from which we are able to observe or understand an object or a scene. Hence, images are the very basic and raw representation of observations, and we would like to record, store, share, and propagate them to the world for various purposes.

While high-quality images are expected in most applications, the imaging process, similar to other sensing techniques, is not perfect. Uncertainty occurs in the measurement of signals, exhibiting blurry, noisy, and other degradation effects in the recorded images. The information of objects and scenes is inevitably distorted, or even lost, the result being that the interesting parts of images cannot convey valuable information to us. Hence, it is necessary to recover the disturbed details as much as possible, such that the underlying scene in an image can be observed clearly [55].

The recent development of advanced imaging devices enables us to overcome some of the imperfect imaging problems from a hardware perspective, *e.g.* by improving the sensors [120] or by stabilising the devices [58, 117]. In spite of this, image processing techniques are still necessary to enhance the quality of the

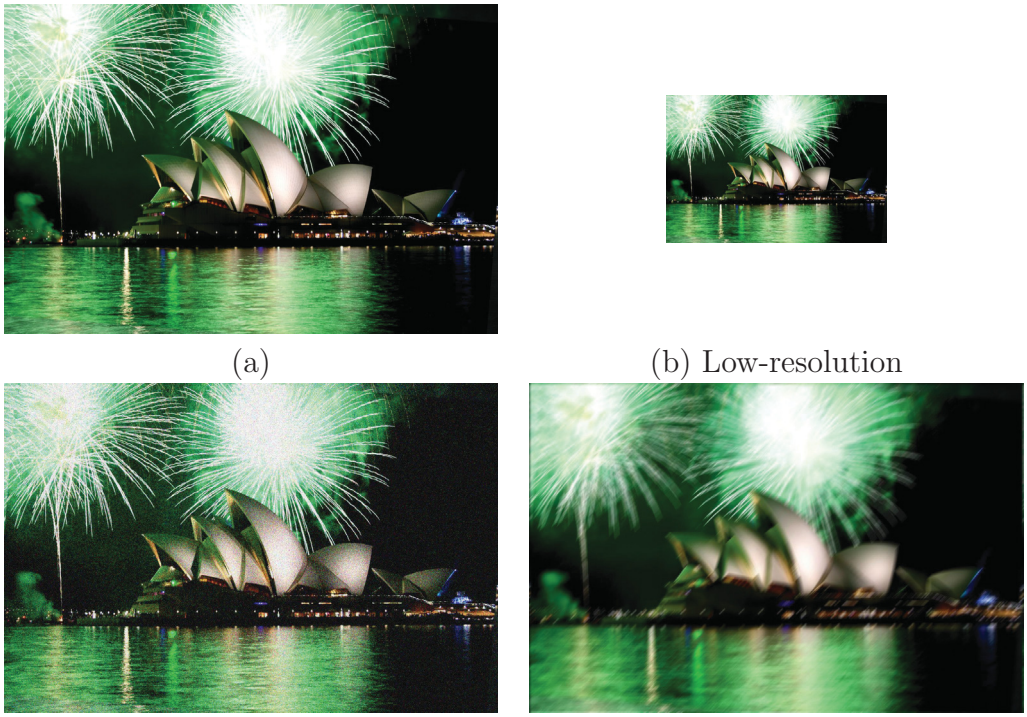
recorded image from a mathematical or a software engineering perspective. Image quality enhancement, which is also referred to as image restoration, typically imposes prior knowledge of the high-quality images that we want to see, and infers high-quality images through complex mathematical derivation or numerical optimisation [18, 54, 201].

The rest of this chapter will consider the background to this problem, how and why we want to enhance image quality, the motivation of this study, as well as a summary of the contributions that have already been made in this thesis.

1.1 Sources of Image Quality Degradations

Photography is a common interest for the majority of people who have access to a camera, particularly when they are on holiday, traveling, or at a social gathering. Even with an advanced device in hand, users can often be disappointed that a significant protion of the captured images may end up being degraded by various factors. For example, when shooting a scene at a long distance, most objects in the image are possibly at a low resolution. While a zoom-in operation can be applied to increase the resolution of local image parts, noise is introduced. An incorrect ISO setting of the camera can also induce a noisy image, which regularly happens when photographers are inexperienced. In addition, in dim light, the amount of available light becomes a restriction for the clarity of the produced image. There is a trade-off between noise and sharpness: short exposure time can result in a sharp image, but graininess is easily observed due to insufficient light; long exposure time permits more light to enter the lens, inducing high signal-to-noise ratio, but movements of the camera and the scene blur details (see Fig. 1.1).

Images coming from other areas, such as remote sensing [130] and medical imaging [100], are also very important resources for subsequent analysis tasks. For instance, remote sensing techniques are used to monitor changes to the earth's surface from space. But during the sensing process through the atmosphere and transmission through telecommunication, the captured images are often degraded by factors such as improper focusing, atmospheric scattering, target motion, and the bandwidth of the telecommunication channel. Medical imaging suffers from the issues caused by the heterogeneous composition of the medium in the body,



(a) (b) Low-resolution

(c) Noisy (d) Blurry

Figure 1.1: Examples of photography. (a) The image we want to see. (b)(c)(d) The images we may capture.

which disturbs the phase and amplitude of the wave as well as its reflections. The resultant images inevitably exhibit noisy, low-resolution and low-contrast effects, imposing difficulties on the disease diagnosis and the follow-up procedure (see Fig. 1.2).

Of course, in each application, the images could be retaken in the hope that the next exposure or sensing will be of high quality, but very often, they are of some unique event that could only be captured once, or the degradation cannot be bypassed due to intrinsic hardware limitations. Hence, it would be more beneficial if the unsightly disruptors could be removed and the details captured in fleeting moments are recovered.

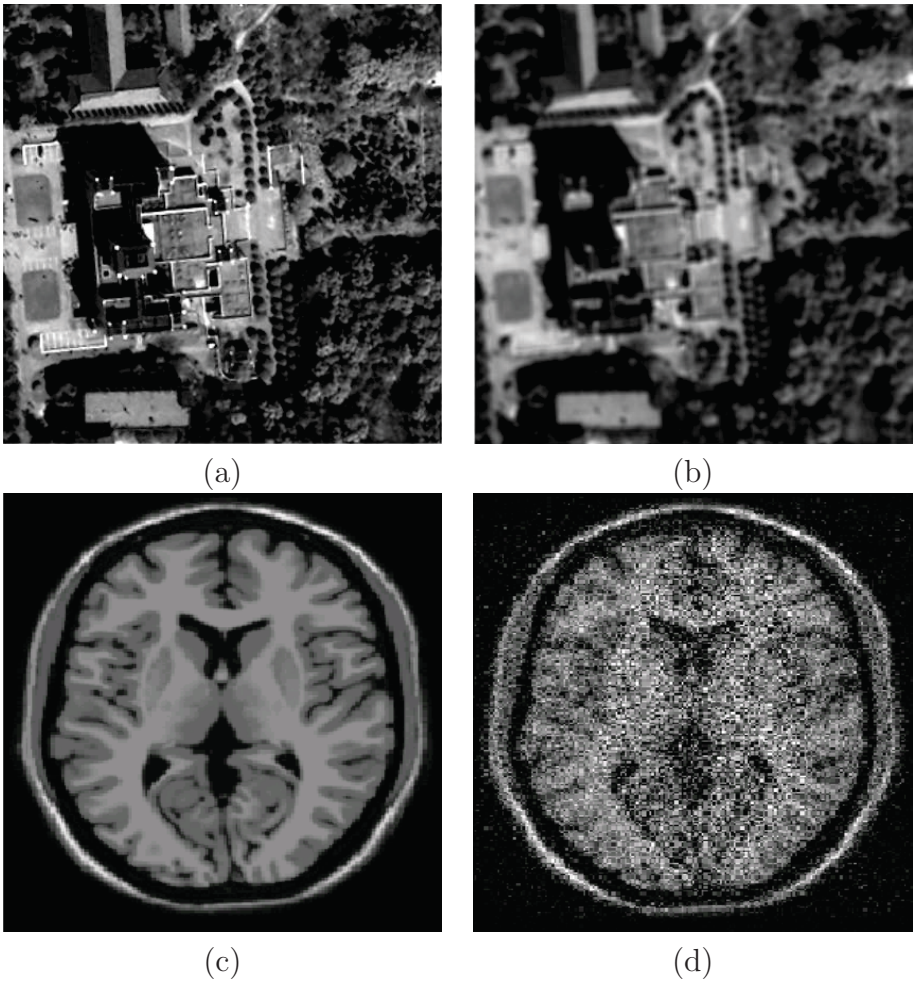


Figure 1.2: Examples of remote sensing and medical imaging. (a)(c) The images that are expected. (b)(d) The images that are degraded.

1.2 Image Formulation and Restoration

Image formation encompasses the radiometric and geometric processes by which a 3D world scene is projected onto a 2D focal plane. In a typical camera system [1], light rays passing through a lens's finite aperture are concentrated onto the focal point. This process can be modelled as a concatenation of the perspective projection and the geometric distortion. Due to the non-linear sensor response,

[1]Different imaging systems correspond to different image formulation models. Here, we start with the camera system and end with the most common formulation for other systems: see Eq. (1.2).

the photons are transformed into an analog image, from which the final digital image is formed by digitization [33].

Mathematically, the above process can be formulated as

$$y = \mathcal{S}(f(\mathcal{D}(\mathcal{P}(s) * h_{ex}) * h_{in})) + n, \tag{1.1}$$

where y is the observed blurry image plane, s is the real planar scene, $\mathcal{P}$ denotes the perspective projection, $\mathcal{D}$ is the geometric distortion operator, $f(\cdot)$ denotes the nonlinear camera response function that maps the scene irradiance to intensity, h_{ex} is the extrinsic blur kernels caused by external factors such as motion, h_{in} denotes the blur kernels determined by intrinsic elements such as imperfect focusing, $*$ is the mathematical operation of convolution, $\mathcal{S}$ denotes the sampling operator due to the sensor array, and n models the noise.

The above process explicitly describes the mechanism of image formation. But for the tasks in which we are interested, the goal is the recovery of a latent high-quality image with no degradations, rather than the geometry of the real scene. Hence, focusing on the image plane, a general formulation that is widely used in the image restoration community is written as:

$$y = \mathcal{R}(h * x) + n, \tag{1.2}$$

where x is the latent image, h is an approximated blur kernel combining h_{ex} and h_{in}, and $\mathcal{R}$ is the down-sampling operator induced from $\mathcal{D}(\cdot)$, $\mathcal{P}(\cdot)$, and $\mathcal{S}(\cdot)$. This equation encompasses the noisy, blurry, and low-resolution cases. Specifically, when a noisy image is observed, it is assumed that $\mathcal{R}$ is an identity function and h is an identity kernel, inducing the formulation:

$$y = x + n; \tag{1.3}$$

when a blurry image is captured, it is assumed that $\mathcal{R}$ is an identity function, inducing

$$y = h * x + n; \tag{1.4}$$

and when a low-resolution is acquired, it is generally assumed that h is an identity

kernel and no noise is added, inducing

$$y = \mathcal{R}(x). \tag{1.5}$$

Image restoration attempts to restore a noisy, blurry, and/or low-resolution observed image to produce a better representation of the latent image. As previously discussed, the uncertainty occurring during the measurement of the image comes from $\mathcal{R}$, h, and n. This makes each of the image restoration tasks (*i.e.* image denoising, image deblurring, and image super-resolution) ill-posed since each of the degradation operators can cause the image details to be lost. Thus, the key point in solving these problems is to incorporate appropriate prior knowledge about the latent image and the type of degradation operator into the restoration process.

In natural image denoising [129], the noise term is usually assumed to follow a Gaussian distribution, which is also considered in our work. Other types of noise include Poisson noise, impulse noise, and speckle noise. Note that Eq. (1.3) is only suitable to describe Gaussian noise which is additive and signal-uncorrelated, whereas other noises are beyond its scope. This problem is difficult since the degradation (*i.e.* the noise) is random, meaning that there can be multiple images generating the noisy observation. To arrive at a solution, we analyze from the Bayesian perspective, obtaining three fashions of formulation, including *maximum a posteriori*, *maximum likelihood*, and *minimum mean square error*. These formulating manners have also been utilized in other restoration tasks such as super-resolution and deblurring. Here, we focus on the *maximum a posteriori* framework to introduce these tasks, since it is the most popular among the others. In image denoising, we consider maximising the posterior probability which is formulated as

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \propto p(y|x)p(x), \tag{1.6}$$

where $p(y|x)$ is the noise assumption, and $p(x)$ encodes prior knowledge about x.

Image deblurring is somewhat a complex case [95]. Conventionally, the problem seeks an estimation of the latent image assuming that the blur is known,

which is termed non-blind image deblurring. In contrast, the blind deblurring problem, which is more practical, means that the blur kernel is unknown, and the task therefore becomes estimating the sharp image and/or the kernel from the degraded observation. The ill-posed condition is unavoidable in either case. In the former, the observed blurry image does not uniquely and stably determine the latent image due to the ill-conditioned nature of the blur operator. That is, if the assumed/given blur kernel and the true kernel are slightly mismatched, or if the blurry image is also corrupted by noise, the recovered image may be much worse than the underlying sharp image. In the latter, even if the blur operator is not ill-conditioned, the deblurring problem will still be inherently ill-posed since there is an infinite set of image-blur pairs that can synthesise the blurry observation. On the other hand, the blur kernel is often assumed to be spatially invariant in the research, but a more practical case is that motion or other factors induce spatially variant blur in the observed image. The resultant restoration problem becomes more difficult since the number of unknown variables in the problem is significantly increased. The objective function for the blind case induced from Eq. (1.4) can again be written from the Bayesian perspective:

$$p(x, h|y) = \frac{p(y|x, h)p(x)p(h)}{p(y)} \propto p(y|x, h)p(x)p(h), \tag{1.7}$$

where $p(x)$ and $p(h)$ are the prior knowledge about the latent image and the kernel, respectively (in the non-blind case, h is known and thus $p(h)$ can be omitted).

When an image is corrupted by noise or blur, we can say that the signal of the scene is *disturbed*, which means that almost all incoming photons from the scene have been captured by the sensor, but they were disturbed. In contrast, in the low-resolution case, the signal of distant objects (which are small in the image) is *partially lost*, since the light rays from distant objects are scattered, and the resolution of the sensor is limited. Image super-resolution is then to generate the lost details based on the observed image, which is difficult, especially when the scale factor is large [166]. Actually, as the scale factor increases, there are more and more high-resolution images that can generate the low-resolution observation, and thus the task exhibits an increasing level of difficulty. Such a phenomenon

inspires different ways to handle large and small scale factors. Overall, the problem is to find reasonable solutions that make sense according to certain criteria. The restoration process is typically performed via an interpolation function or a discriminative mapping function, which can be viewed as an inverse function of $\mathcal{R}$.

1.3 Philosophy of Image Quality Enhancement

Image quality is one of the most important factors in almost all imaging systems. While a high quality is expected, we may expect to resort to a better observation procedure and a more advanced device to enhance the quality. But there still exist physical limits, such as photon diffraction and scene motion, which cause the captured images to be sub-optimal. This makes it necessary to digitally improve the quality of an image through post-work.

Photography is always transitory. An event which occurs at a particular time generally cannot reoccur, thus we cannot retake an image to obtain a better photo. Even though it is possible that a similar event may occur through intentional guidance, users would most likely be again disappointed with the results. Sometimes re-imaging may incur great risk: for example, it is not recommended that an X-ray image be retaken for a patient when considering a health issue. Hence, digitally enhancing the quality of an existing image is the only feasible option to acquire a satisfactory image.

After users obtain a degraded image, they may want to change their observation process and upgrade their imaging devices: using more expensive cameras with lenses of larger aperture, allowing for faster shutter speed; using optical image stabilisers allowing for the compensation of camera motion; using flash to provide extra illumination. However, these options can even be ineffective in some cases, *e.g.* when objects move. In low-light conditions, the illumination from a flash usually provides unnatural light effects and washes out the object's appearance due to the location of the light source. Also, a flash may be prohibitive in some specific scenarios. Therefore, to obtain a natural looking photo, post-processing is often required.

Another consideration in seeking a solution for degraded images is cost. Ad-

vanced lenses, sensing equipments, and hardware corrections are expensive, and sometimes are not suitable for inexperienced users due to complex and non-user friendly operations. But the recent developments in computational resources make it much cheaper to perform a small set of computations on a micro-chip, thus opening the door to the application of increasingly sophisticated models.

Digital image restoration is a sound way to gain maximal useful information from the available signals. The restoration models are mostly inspired by the corresponding imaging processes. A conventional view states that the imaging device is designed to gather the required information and signal processing is an afterthought to compensate for the deficiencies in the measurements. But in fact, hardware and software are developed simultaneously. The restoration algorithms can become a part of the imaging system, motivating us to design hardware and software in a harmonious fashion. Such a direction could provide many advantages and inspiration to the future of imaging.

1.4 Motivations

Most of the existing research on image quality enhancement is either based on the computationally expensive patch-level operations which involve searching similar blocks, or based on complex iterative Bayesian inference processes that require a large amount of memory/time to output the restoration results [34, 40, 48, 201]. Even though the best performance record is being rewritten, these methods are far from being integrated into devices which have limited computing and memory units. The limitations are nowadays being addressed by incorporating the deep learning framework which is a powerful modelling tool. Key considerations of employing deep models are that nonlinear mappings and hierarchical abstractions can handle complex image local structures (whereas conventional methods generally assume a linear subspace among similar blocks) [6,136]; and discriminative modelling allows the methods to produce results with one round of forward propagation, which is much more efficient than the conventional Bayesian inference. These advantages motivate us to develop deep learning-based image quality enhancement techniques for real-time applications.

The deep learning paradigm formulates restoration problems in a discrimina-

tive manner. With a complex and very deep architecture which contains a large amount of parameters, this approach can model uncertainty in a wide range (for example, different noise levels, different scale factors, and different blur kernels). A very important point for the success of deep learning is the use of massive training data that enables the model parameters to be thoroughly trained. Even though the non-convex optimisation issue exists, most deep models can be learned with a careful design of the training process, to reach a local optimum which empirically shows promising performance in different tasks.

1.5 Summary of Contributions

To make use of the deep learning framework, it will be necessary to investigate: 1) how the degradations in images could affect the mechanism of deep models; 2) what properties of the designed architectures could critically influence their performance in restoration tasks; and 3) what factors may increase the difficulty of training deep models. In this thesis, we develop appropriate models that conquer or bypass the previous issues. The contributions are summarised as follows:

Chapter 2 provides a general review of the current research on image restoration, focusing on the state-of-the-art problem formulation and optimisation methods. Specifically, the recent developments on image denoising, super-resolution, and deblurring are discussed.

Chapter 3 describes a method motivated by a pivotal property of the human perceptual system, which states that similar visual cues can stimulate the same neurone to induce similar neurological signals. Image degradations can result in the fact that similar local structures in images exhibit dissimilar observations. While conventional neural networks do not consider this important property, we develop the (stacked) non-local auto-encoder which exploits self-similar information in natural images to enhance the stability of signal propagation in the network. It is proposed that similar structures should induce similar network propagation. This is achieved by constraining the difference between the hidden representations of non-local similar image blocks during training. By applying the proposed model to image restoration, we then develop a "collaborative stabilisation" step to further rectify forward propagation.

Chapter 4 addresses the question as to which factor, receptive field size or model depth, is more critical when applying deep models to image quality enhancement. To arrive at the answer, we focus on the single image super-resolution task and propose a strategy based on dilated convolution to investigate how the two factors affect performance. Our findings from exhaustive investigations suggest that single image super-resolution is more sensitive to changes in receptive field size than to model depth variations, and that model depth must be congruent with the receptive field size to produce improved performance. These findings inspire us to design a shallower architecture which can save computational and memory cost while preserving comparable effectiveness with respect to a much deeper architecture.

Chapter 5 studies the general non-blind image deconvolution problem. It is observed in practice that by using existing deconvolution techniques, the residual between a sharp image and the estimation is highly dependent on both the sharp image and the noise. These techniques require the construction of different restoration models for different blur kernels and noises, inducing low computational efficiency or highly redundant model parameters. Thus, for general purposes, we propose a method by designing a very deep convolutional neural network which can handle different kernels and noises, while preserving high effectiveness and efficiency. Instead of directly outputting the deconvolved results, the model predicts the residual between a pre-deconvolved image and the corresponding sharp image, which can make the training easier and results in suppressed artifacts in the restored image.

Chapter 2

Literature Review

Image quality enhancement has a long history that begins with the invention of digital imaging techniques. Through exhaustive research on this topic, great success has been achieved to date, enabling us to enjoy superb and beautiful photos captured by oneself or crawled from media-sharing websites. The previous efforts have largely inspired the achievements of this thesis. Hence, before articulating the details of our work, in this chapter, we provide the literature review of different enhancement tasks, including image denoising, image super-resolution, and image deblurring. Since deep learning is the basic framework of our work, we give a review on the recent progress of deep learning and introduce the basic concepts of auto-encoders and convolutional neural networks at the end of this chapter.

2.1 Image Denoising

A common choice in the research community of image denoising is to assume that the noise is the additive Gaussian noise because this can simulate many signal-uncorrelated noises in real applications, as modelled in Eq. (1.3). Typically, the Gaussian distribution has a zero mean value and a standard derivation of σ. While the reviewed methods are mostly designed to handle such type of noise, they have also inspired work on other types of noise.

To remove noise from an image, the motivations for designing algorithms have transferred from how to operate the pixels in the image, to how to correlate

the on-hand noisy image with other clean images. This shift of motivation can help us to recognise the existing denoising methods in three categories: spatial domain-based, transform domain-based, and learning-based approaches.

2.1.1 Spatial domain-based approaches

Spatial domain-based methods are operated on the origin pixels and consider the spatial correlations among them. Specifically, a pixel in an image is highly correlated with its surrounding pixels, meaning that this pixel can be represented as a weighted summation of the surrounding ones. This idea has inspired a lot of local filtering algorithms. As a counterpart, a pixel can also be correlated with a distant pixel considering the fact that their local structures (*i.e.* the patches in which the pixels are centred) are similar to each other. This is an important property of natural images, termed redundant patch recurrence or self-similarity, which is the key motivation of nonlocal filters. From these discussions, we know that spatial domain-based methods can be further classified into local filters and nonlocal filters.

Given a pixel, local filters typically determine the weights of the surrounding pixels according to their distance with respect to the central pixel. The weighting scheme is the most important factor in this class of methods. The conventional filters are average filter, median filter, and Gaussian filter, etc. [55], which perform very efficiently but will produce blurred edges in the results due to the isotropic property of the filter structure. Compensating for the drawbacks of these filters, anisotropic filters are designed to smooth images by considering the structural similarity between the surrounding and the central pixel [124, 134, 151, 163]. Such a consideration encourages the results to exhibit suppressed blurry effects and sharp edges. To encourage more adaptivity to natural images, the filters can also be determined through a learning process on a large image dataset [146]. Given different image local structures, each can be restored by using a certain pre-trained filter. The whole process contains a structure classification step that determines the type of a local structure, and a filtering step in which the filter is selected from pre-trained ones, according to the classification result.

Similar to local case, nonlocal ones also restore a pixel by computing a weighted

summation over a set of pixels. The differences are that the pixels in the set can not only be around the target pixel but also be distant to it, and that the weight for a pixel is determined by the similarity between two patches centred on that pixel and the target pixel, rather than by the similarity between two individual pixels [15]. Thus, if the local structures around two pixels are more similar, these pixels contribute more to each other; otherwise, the contribution is less. This enables an accurate estimation of weights, producing more pleasing restoration results compared with the local filters. The success of nonlocal filters can be further explained by the fact that the noise is generally assumed to be independently identically distributed and can easily corrupt each pixel, but the structure in a local area (say a patch) is observable and identifiable when the noise is not too severe. So, measuring the similarity between two local structures is more effective than measuring the similarity between two pixels. The performance of nonlocal filters can be further improved by searching more similarity information, designing an iterative denoising process, and changing the distance measurement [56, 200].

Following the nonlocal ideas, regularizations and patch priors have been proposed to encourage performance improvement. For example, Zoran and Weiss [201] formulated the prior of clean image patches with a Gaussian mixture model, and a noisy patch was restored by maximizing the expected patch log-likelihood (EPLL). Gu et al. [59] proposed the weighted nuclear norm minimization (WNNM) algorithm which formulated the low rank problem by involving a weighted nuclear norm. IN the following, Xie et al. [182] applied a similar weighting scheme to the Schatten p-norm minimization problem, which achieved performance improvement. To obtain more accurate information of self-similarity in clean images, the patch group prior-based method was proposed to learn the explicit non-local models from clean images, which was then employed on noisy images for denoising. Chen et al. presented a new patch clustering method where the patch clusters were formed through the guidance of a learned Gaussian mixture prior.

2.1.2 Transform domain-based approaches

According to the name, transform domain-based methods represent an image with a series of coefficients in terms of a set of bases, and then process the coefficients

for denoising. Sometimes, these methods perform better than the spatial domain-based ones because of an important property which states that the transformation can effectively separate the noise and the original signal. Conventional and well-known approaches, such as Fourier transform and discrete cosine transform [55], by which the noise in the transformed domain is located in the high frequency band, can be used. A low-pass filter can be applied to completely remove the noise, however, the signals of textures and edges which are also located in the high frequency band tend to be degraded, exhibiting blurry effects in the restoration results.

The wavelet transform [114] improves the previous transformations in the aspect that the spatial property and the frequency property of an image are simultaneously preserved during transformation. As a result, we obtain two advantages: one is that the low frequency signal and the high frequency signal are separated, which is the same as the previous transformations; and the other is that noise, edges, and textures are spatially separated, and noise possesses smaller coefficients than the others. Hard thresholding, soft thresholding [42, 43] or probabilistic models [129] can then be applied to discard the small coefficients in high frequency bands, accomplishing the goal of denoising.

While the above-mentioned methods are performed locally, an extension to the nonlocal case has also been attempted, namely BM3D [30], which showed a substantial improvement in performance when it was published. Clearly, BM3D has all the advantages that the nonlocal filters possess. We should also analyse why this method can produce such good performance by using wavelet transform or discrete cosine transform. A strong assumption in previous transform domain-based methods is that the signal is stationary. This guarantees that the filtering operation in the transform domain is theoretically reasonable. However, violated cases may occur in practice, and non-stationary signals are observed, causing these methods to be sub-optimal. From a different view, BM3D actively searches similar patches from images and stacks them into a 3D cube, such that at each 2D coordinate, the signal along the third dimension is stationary. In this way, transformation-based denoising can be performed more effectively. A method similar to BM3D is LPG-PCA [196] which applies principal component analysis across similar patches and then processes the coefficients in the PCA domain.

2.1.3 Learning-based approaches

The learning-based methods employ modern machine learning techniques and benefit from the abundant high-quality images acquirable from the Internet. The main idea of learning is to optimise a module of the designed model, based on training data. An example is dictionary learning. Image patches can be represented with a series of coefficients on a set of bases which forms a redundant dictionary. For the task of denoising, the dictionary is expected to express the general structures of natural images, such that the clean image patches are well estimated on it. Thus, the dictionary is an important part which is typically learned from a set of high-quality training data or the to-be-processed degraded image, as in KSVD [45], LSSC [113], and CSR [38]. Another example is deep learning, in which the goal is to learn a restoration function in a discriminative fashion. Initial attempts to apply convolutional neural networks (CNNs) and stacked auto-encoders to natural image denoising [80, 180] are promising, which indicate that these deep models can provide comparable or even superior performance to the conventional wavelet or Markov random filed-based denoising methods. A remarkable study by Burger et al. [16] demonstrated that neural network-based models were comparable to state-of-the-art methods such as BM3D [30]. Further improvements [17] were achieved by endowing the deep neural networks with the images restored by BM3D [30] and EPLL [201]. Agostinelli et al. [1] investigated the possibility of constructing multi-column stacked auto-encoders to tackle different noise types. DnCNN [194] employed the residual learning strategy to train the deep CNN models and achieved state-of-the-art performance. This strategy enabled the model to handle different levels of noise and different types of degradation that appeared in a single image. These studies show that deep architecture properties, namely their strong learning capability and high representational capacity, are well suited for image denoising, and can substantially improve the performance of image denoising through the utilization of redundant high-quality images acquirable from Internet.

2.2 Image Super-resolution

Super-resolution aims to construct a high-resolution image from one or more observed low-resolution images, thereby recovering the high frequency details lost during the imaging process. Our work mainly focuses on the task of single image super-resolution in which very limited information about the image details is provided, making the problem very difficult. Comprehensive review articles include [121] and [185].

The most classic methods are interpolation-based, which estimate the pixels in high-resolution grids through a parametric interpolation function. These include Bilinear interpolation, Bicubic interpolation, nearest neighbour interpolation, fixed-function kernels, and structure-adaptive kernels [86, 106, 197]. Even though these interpolation functions can be executed very quickly, the restoration results are far from satisfactory because blur effects and zigzag artifacts can be easily generated. This is because the models have limited receptive field size and low complexity. More powerful methods are either reconstruction-based or learning-based, which are reviewed in the following.

2.2.1 Reconstruction-based super-resolution

As introduced previously, the ill-posed issue of super-resolution is caused by information loss. The reconstruction-based methods address this issue by incorporating certain a priori to generate high-frequency components. Thus, the type of a priori is the key factor that motivates different algorithms.

One type of a priori regards gradient profiles. It is known that the shape statistics of gradient profiles in natural images are robust against variation in image resolution. This means that the statistics obtained from a low-resolution image can be used in the restoration process as a constraint on the high-resolution image. Specifically, to implement this idea, the gradient profile-based methods [154, 155] assume the distribution of gradient profiles to be the generalised Gaussian distribution which is parametric. The similarity between the shape statistics of the low-resolution and high-resolution images is measured from external data, which is then employed as the constraint when restoring a degraded image.

The Fields of Experts [61, 133] is a priori to approximate the heavy non-

Gaussian statistics of natural images. In this case, a series of bases is derived from natural images with each acting as an expert, expressing a certain local structure. An image can then be represented as a set of coefficients on these bases. By reasonably constraining the coefficients, we could expect that the reconstructed high-resolution image has consistent properties with other high-resolution natural images.

Class-based a prioris are also designed for super-resolution, where the key idea is used in hallucination-based algorithms [47,156]. Specifically, we can use primal sketches to determine the shape of a class of objects. By using the a priori of primal sketches, the reconstructed object should have a consistent sketch with the same class of objects. Regarding this, the hallucination algorithm is applied to the primitives (*e.g.* corners, edges, and ridges) to generate the high-frequency details of these parts, while Bicubic interpolation is used on the non-primitive parts of the image.

These reconstruction-based methods generally promote sharper edges and suppress more aliasing artifacts than the interpolation-based methods, owing to the appropriate priors and iterative inference processes. However, the complex inference processes induce high computational cost, limiting the feasibility in real-time applications.

2.2.2 Learning-based super-resolution

The learning-based methods are inspired by machine learning techniques and involve a training step in which the relationship between high-resolution images and their low-resolution counterparts is learned from an image dataset. Through such a data-driven strategy, the learned knowledge can be employed as a priori for the reconstruction. Considering how to use the a priori, we can organize different types of learning-based algorithms, as discussed in the following.

A belief network is a type of learning strategy which can be specified as a Markov network or a tree structure. The Markov network [49–52] analyses the images with patch representation. The corresponding high-resolution and low-resolution patches are related to each other with an observation function, which defines how well one matches the other. The neighbour patches in the recon-

structed image are correlated by a transition function. After the parameters of the functions have been well trained, the model restores the high-resolution image by using the belief propagation algorithm. Similar to this system, a mixture of tree structures [153] is also defined in the belief network for learning the relationship between high-resolution and low-resolution images.

Manifold learning-based methods [11, 20, 22] assume that high-resolution images form a manifold with a similar local geometry to that formed by low-resolution images. So the relationship between points on the low-resolution manifold can be directly applied to the high-resolution manifold in order to undertake restoration. However, the assumption of similar manifolds is somewhat too restricted, and in many cases cannot be fulfilled. To handle this issue, it is proposed to find a common manifold for the high- and low-resolution images by learning two explicit mapping functions [104, 105].

The sparse coding-based methods [41, 53, 65, 170, 187–189, 195] are based on the theory that a signal can be represented by a sparse code on an over-complete dictionary. In the task of super-resolution, it is possible to recover the linear relationship between the sparse codes of high- and low- resolution images, which can produce promising restoration results. Specifically, by dividing images into patches, a high-resolution (low-resolution) dictionary can be obtained from the high-resolution (low-resolution) patches. Given the dictionaries, the sparse code of a low-resolution patch can be inferred, which is generally assumed to be identical, linearly, or non-linearly related to the code of its high-resolution counterpart. the types of the mapping function include simple functions [186], kernel regression [90], support vector regression [123], and anchored neighbourhood regression [164, 165].

While the previous models are indeed shallow models with limited representational capacity, it has been shown that stacking shallow models into a deep one can substantially increase the performance of super-resolution, which relates to deep learning. For example, the very recent works on CNN produced surprising improvements on SISR performance, particularly in terms of the peak signal-to-noise ratio (PSNR). Following the initial CNN work of Dong et al. [35, 36], an investigation on using very deep convolutional networks for SISR was conducted by Kim et al. [88], who proposed to train a 20-layer CNN model via residual

learning and showed the state-of-the-art PSNR values. The same group also presented a deeply-recursive convolutional network [89] that allows for long-range pixel dependencies by using small number of model parameters. Wang et al. [173] proposed to employ self examples of multiple scales to fine-tune a pre-trained convolutional auto-encoder in the SR task. Similar to the architecture of [88], Wang et al. [172] developed a combined deep and shallow network, where the shallow network can be regarded as learning a pre-super-resolved image from the LR input, while the deep network learns the residuals as in [88]. To accelerate the SR computation, Shi et al. [149] and Dong et al. [37] designed networks in which most of the computation occurs in the LR space and the up-scaling operation is only performed in the last layer of the networks. Besides the above CNN-related works, the studies on other feed-forward neural networks were conducted, such auto-encoders [29, 171, 191] and sparse coding-based networks [110, 174].

A question that recently attracts the attention of SISR community states that: is PSNR, or equivalently mean square error (MSE), a good objective function for SISR in terms of perceptual satisfaction, when the image is heavily down-sampled? According to the analysis by Ledig et al. [99], a LR image patch can be generated from different HR image patches which reside in a low-dimensional natural image manifold. The MSE-based solution is a pixel-wise average of the possible HR patches on the manifold, thus exhibiting blurry or over-smoothing effects, and lacking high-frequency details. To address this issue, Johnson et al. [82] proposed a perceptual loss which measures the MSE between the VGG feature maps of the SR result and the ground truth. By incorporating this loss and an adversarial loss, Ledig et al. [99] developed a generative adversarial network which can recover photo-realistic textures. Using similar ideas, inference models can be applied to the statistics of CNN feature maps, such that the statistics of SR solutions and that of natural images are as close as possible (refer to the works by Bruna et al. [14] and by Sønderby et al. [152]).

2.3 Image Deblurring

The ill-posed condition is the most severe problem in image deblurring. In the non-blind case, the observed blurry image does not uniquely and stably determine

a sharp image due to the ill-conditioned nature of the blur operator [9]. This means that if the assumed/given blur kernel and the true kernel are slightly mismatched, or if the blurry image is also corrupted by noise, the recovered image may be much worse than the underlying sharp image. In the blind case, even if the blur operator is not ill-conditioned, the deblurring problem will still be inherently ill-posed since there is an infinite set of image-blur pairs that can synthesise the observed blurry image. A more difficult case occurs when the blur is spatially variant, since the parameters of the blur kernels are significantly increased.

Contemporary researchers have been devoted to developing new models and new prototypes or improving the efficiency of optimisation methods, to deal with ill-posedness. In the rest of this section, we first review the deblurring methods from the methodology perspective, and then introduce the existing methods of modelling spatially varying blur.

2.3.1 Methodology on deblurring

From a model construction perspective, most methods for handling spatially invariant blur can be grouped into one of the following four categories: Bayesian inference framework, variational methods, sparse representation-based methods, and learning-based methods.

In the Bayesian inference framework, priors are introduced to impose uncertainty attributes on either the unknown sharp image or the unknown blur kernel, or both. This operation is intended to reduce the volume of the search space, where the problem's solution lies, to suppress ill-posedness. Traditionally, the maximum a posteriori (MAP) and the minimum mean square error (MMSE) are widely used in estimating the optimal solutions of the sharp image and the kernel. For example, the Richardson-Lucy method [111, 131] assumes that the noise follows a Poisson distribution, while Wiener Deconvolution [178] imposes Gaussian assumption on the noise term. To improve the performance of deconvolution and suppress the artifacts, various priors are proposed, which characterise the properties of sharp images. It is commonly agreed that for a natural image, the gradients of the sharp image tend to follow a heavy-tailed distribution, meaning

that the distribution of gradients has most of its mass on small values but assigns significantly less probability to large values than a Gaussian distribution [48]. Thus natural images often contain large regions of constant intensity or gentle intensity whose gradients are interrupted by occasional large changes at edges or occlusion boundaries. At this regard, a classic method to describe the gradients is to impose a Laplace distribution on the magnitude of the gradients. Besides the regularizers on image gradients, Shaked et al. [145] saddled the generalized Gaussian prior on the representing coefficients in a linear transformed domain of the sharp image. A notable property of images is that the patches of a natural image can be constrained to be close to a low dimensional manifold [122, 126]. Sun et al. [158] exploited context-appropriate image priors that benefit non-blind deconvolution. For a better fitness, Gaussian mixture model [48] and Gaussian scale model [19] have shown improved performances. However, a notable issue in the Bayesian inference framework is that using MAP to simultaneously estimate the clean image x and the blur kernel h (denoted as $\text{MAP}_{x,h}$) under a sparse derivative prior usually gives a blurry result, failing to recover a sharp result. This observation is known as the no-blur explanation or delta effect of MAP. To address this issue, Levin et al. [102, 103] proposed a scheme (denoted as MAP_h) whereby using MAP to estimate h by marginalising over x can arbitrarily reach the true kernel. Regarding this, variational Bayesian inference has been employed to tackle the marginalisation problem [4, 48, 103].

Variational methods render the solution unique and stable through the incorporation of regularisation techniques whose role is similar to the a prioris in Bayesian inference, *i.e.* regularising the solution into a constraint space. An ideal regularizer is the ℓ_0 constraint on gradients, since it is commonly agreed that the gradients of a sharp image should be sparse. Due to the lack of derivative information everywhere, ℓ_0 is usually approximated by ℓ_1 which, however, fails to preserve or recover the energy of the original gradients. Various techniques have been developed for approximating ℓ_0 more closely, such as normalised ℓ_1 [92] and unnatural ℓ_0 [184]. Jung et al. [84] derived a nonlocal Mumford-Shah (MS) regularizer for image restoration. Another concern with variational methods is the formulations for modelling the variational image restoration problem. Typically, the formulations can be grouped as either analysis formulation or synthesis

formulation, each of which has merits and shortcomings in specific tasks. Furthermore, a combination of these two formulations is recommended to facilitate more promising performance, such as the block-matching 3-D (BM3D) modelling [31] and balanced formulation [147, 181].

Sparse representation benefits from the fact that natural images are intrinsically sparse in some domains. The sparse property of the natural images in these domains is well suited to alleviate the ill-posed condition of the deblurring problem. The focus of sparse representation methods includes the development of the dictionaries and the smoothness of the results. For dictionary learning, task-specific dictionaries [2], data-driven dictionaries [125], and structured dictionaries [39–41] are proven to be advanced in the recovery performance. Regarding smooth enhancement, integrating the local regularisations and the nonlocal regularisations into sparse representation modelling is an effective way to suppress the artifacts and enhance the smoothness [41, 112].

As in image denoising and super-resolution, learning-based methods are proposed for the deblurring task. The motivations here can be summarised in two groups. In the first group, the methods [83, 122, 157, 201] aim to learn a subspace where the sharp image may be located. By extracting the local patterns from multiple sharp images, a subspace can be constructed, in which the target image and the sharp images share similar details and thus details in the target image can be accurately represented. In the second group, the goal is to learn a restoring function which recovers the sharp target image from its blurry version [139, 142]. In this case, multiple sharp images together with their blurry counterparts are generally utilized to train the parameters of the restoring function. For example, Schmidt et al. [139] employed the regression tree field (RTF) to model a nonlinear regressor that specifies the local deblurring parameters. Before deblurring an image, RTFs are learned by maximising a peak signal-to-noise ratio-based loss function. Rather than directly learning a deblurring function, Schuler et al. [142] estimated the deblurred image by using direct deconvolution [69] in the Fourier domain, and then incorporated the multilayer perceptron (MLP) to remove the artifacts of that deblurred image. Xu et al. [183] took advantages of kernel separability and developed an initialisation method for the convolutional layers in convolutional neural networks (CNN). They formulated the image non-blind de-

convolution as an end-to-end learning process without pre- or post-processing.

2.3.2 Modelling of spatially variant blur

Spatially variant deblurring handles the case where the blur kernel is spatially variant, which is quite normal in many real applications. Homography-based modelling and region-based methods are proposed for this problem. Since spatially variant blur cannot be modelled by a single blur kernel with limited support, researchers usually approximate the blurry effect by using a union of multiple kernels or homographies. Due to the growing number of unknowns, this type of method leads to a worse situation in the inverse problem. To overcome this issue, homography-based methods derive the spatially variant kernel as an integration of "temporal" homographies whose continuity is further constrained according to the properties of the blur effects in the image. Since the integration involves infinite instantiations of homographies, the resultant problem becomes ill-posed. To handle this, one option is to apply time discretisation such that the homography is approximately consistent in each period [27, 162], while another option is to decompose the whole homography into a set of basic homographies [21, 62, 177]. Since the spatial variant blur kernel implies that the local kernels vary from region to region, or more strictly, from point to point, region-based methods express the blur model in that each region, or ideally, each point is assigned to a local consistent kernel. A notable example is the efficient filter flow (EFF) model proposed by Hirsch et al. [63, 70].

2.4 Deep Learning

The research on neural networks has a long history that begins from 1960s [75], with the publications of shallow neural networks [77–79] and the back-propagation algorithm [108, 109]. Since then, the field of neural networks has obtained a rapid development until 1990s, producing a series of basic network units such as auto-encoders [5], convolutional neural networks [96, 97], recurrent neural networks [71, 176], and restricted Boltzmann machines [127]. The training of those neural networks was based on the back-propagation algorithm, which, however,

was shown to be difficult in training a deep model due to insufficient computing power at that time. Then, the research community shifted the focus on alternative machine learning methods such as kernel machines [141, 167]. While these methods had shown their effectiveness in a certain range of applications, the requirement of high model complexity and data explosion highlight their practical limitations especially in such a big data era. Benefitting from the development of advanced computing devices and the help of hierarchical unsupervised learning, in 2000s, it became possible to learn an outperforming deep neural network and achieve the state-of-the-art performance in various tasks. In this section, we give an introduction about the auto-encoder and the convolutional neural network, which are the basic network units employed in our following work.

2.4.1 Auto-encoder

The basic auto-encoder aims to discover intrinsic structural features in a set of data. It was originally proposed to perform unsupervised leaning in a hierarchical architecture. Specifically, the lowest-level auto-encoder is learned to map input patterns to themselves through a single hidden layer. The obtained feature from the hidden layer is then fed into a higher-level auto-encoder, and so on. In this way, an auto-encoder stack is formed with the hope that the feature in each hidden layer possesses good properties that facilitate subsequent learning. It has been shown that learning a deep neural network with this hierarchical pre-training strategy works better than learning the model by back-propagation without pre-training [5, 67, 128].

In mathematics, given a vectorized input $\mathbf{x} \in \mathbb{R}^d$, the auto-encoder architecture involves an encoder and a decoder. The encoder is realized as a deterministic mapping that transforms $\mathbf{x}$ into a hidden representation $\mathbf{h} \in \mathbb{R}^{d'}$, i.e.,

$$\mathbf{h} = s(\mathbf{W}\mathbf{x} + \mathbf{b}). \tag{2.1}$$

The parameter set includes $\mathbf{W}$, a $d' \times d$ weight matrix, and $\mathbf{b}$, an offset vector of dimensionality d'. $s(\cdot)$ is a non-linear mapping such as the sigmoid function or the hyperbolic tangent function. The decoder reconstructs a d-dimensional vector $\hat{\mathbf{x}}$ based on the resulting hidden representation $\mathbf{h}$. The formulation is

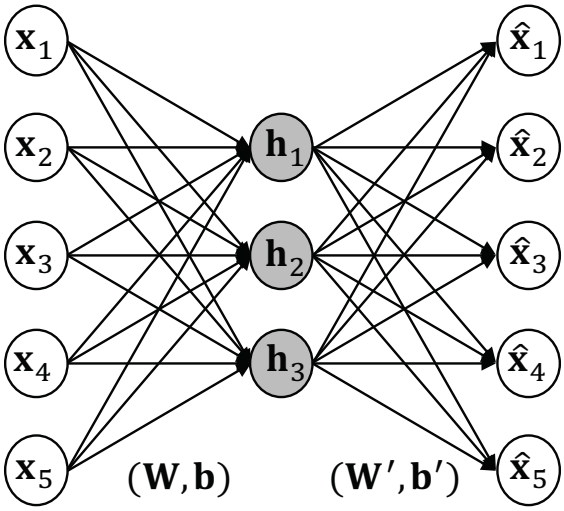


Figure 2.1: A typical architecture of the auto-encoder.

again a non-linear transformation:

$$\hat{\mathbf{x}} = s(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{2.2}$$

or an affine mapping $\hat{\mathbf{x}} = \mathbf{W}'\mathbf{h} + \mathbf{b}'$ with appropriately sized parameters $\mathbf{W}'$ and $\mathbf{b}'$.

In general, $\hat{\mathbf{x}}$ is interpreted as a reconstruction of input $\mathbf{x}$. In the context of image restoration, however, the auto-encoder takes the input of the degraded signal $\mathbf{y}$ (i.e., $\mathbf{x}$ in Eq. (2.1) is replaced with $\mathbf{y}$) and output $\hat{\mathbf{x}}$ to approximate the true signal $\mathbf{x}$. We write $\hat{\mathbf{x}} = f_\theta(\mathbf{y})$, where f_θ is parameterized by $\theta = \{\mathbf{W}, \mathbf{b}, \mathbf{W}', \mathbf{b}'\}$ and encodes the composed forward pass of Eqs. (2.1) and (2.2). This yields an associated reconstruction error for optimization:

$$\begin{aligned} \ell(\theta; \mathbf{x}, \hat{\mathbf{x}}) &= \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 \\ &= \|\mathbf{x} - f_\theta(\mathbf{y})\|_2^2, \end{aligned} \tag{2.3}$$

where the squared error is used for consideration of real-valued $\mathbf{x}$. For binary-valued $\mathbf{x}$, the cross-entropy loss is generally used, which is

$$\ell(\theta; \mathbf{x}, \hat{\mathbf{x}}) = -\sum_j [\mathbf{x}_j \log \hat{\mathbf{x}}_j + (1 - \mathbf{x}_j) \log(1 - \hat{\mathbf{x}}_j)], \tag{2.4}$$

where the subscript j is the index.

2.4.2 Convolutional neural network

Convolutional neural networks (CNNs) are a special family of neural networks that are nowadays popular. They are originally designed to analyze image contents. The advantage of CNNs over auto-encoders and other fully connected neural networks is that CNNs exploit the local spatial structures of an input image, well keeping the spatial relationships of pixels. They do not treat far away input pixels in the same way as those which are close by. This avoids over-complexity during modelling and enables a reliable training of the model parameters. Stacking the basic convolutional neural networks into a deep model works in the same way as stacking classical neural networks, that is, the output from one layer serve as the input for the following one. The training of deep CNNs is through the back-propagation algorithm.

In mathematics, a basic convolutional layer is formulated as

$$f_{i,j}^{l} = \sigma\left(\sum_{t}\sum_{k} w_{t,k} f_{i-t,j-k}^{l-1} + b\right), \tag{2.5}$$

where $f_{i,j}^{l}$ is the output value at (i,j) in the feature map of l-th layer, $w_{t,k}$ is the filter value at position (t,k), b is the shared value given to the filter bias, and $\sigma(\cdot)$ is the activation function providing the non-linearity. In CNNs, ReLU is generally used as the activation function [119]. The filter size $T \times K$ varies in different CNN architectures, and is generally dependent on data. Generally, a large input requires a large filter size to analyze identifiable image structures, while a small input requires a small filter size. But in a stacked deep model, the equivalent filter size of the last layer can be seen as an addition of the filter sizes in all previous layers. This brings a concept of CNNs, which is *receptive field*. It is defined as the field in the input image which is involved in the computation of a unit in a certain hidden layer.

In fact, receptive field, model depth, and non-linearity are three main factors determining the capacity of a deep CNN model. In recent years, different models have been proposed to increase these factors through different designs of the

architectures. For example, after LeNet [98] which has three parameterized layers, a notable deep CNN model is AlexNet [93] that has 7 parameterized layers and is the winner of the ILSVRC 2012 challenge. After that, the GoogleLeNet model [160] was proposed which contained 22 parameterized layers and won the object detection task of the ILSVRC 2014. At the same time, another noteworthy model was VGG [150] which has 16 convolutional layers plus fully connected layers at the end; this model achieved the second-best position in ILSVRC 2014. In the following years, the GoogleLeNet model, also known as Inception, was being updated, producing deeper models including Inception-V2 [76], Inception-V3 [161], and Inception-V4 [159]. An important work that perhaps has significant influence on the deep learning community is Residual Network [64], which can be even stacked into 1000+ layers; this model refreshed the state-of-the-art record of image classification in ILSVRC 2015. While enlarging or strengthening them can substantially improve the model performance in various tasks, the memory cost and computation load can be amplified at the same. Considering the practical usage, a balanced design should be made. An investigation of this problem in the image super-resolution task is conducted in Chapter 4.

Chapter 3

Non-local Auto-encoder with Collaborative Stabilization

Human vision system has the capability of recognising objects from a degraded scene, exhibiting high robustness. This can be explained by that the system can subjectively filter the degradations and process similar visual cues to produce similar neurological signals. Similarly, neural networks are also expected to possess such a property. However, since degradations are caused by the uncertainty during the imaging procedure, they corrupt image structures in a random way, resulting in that similar local structures in images exhibit dissimilar observations. To investigate how this issue influences the performance of neural networks, we empirically find that when two degraded inputs (of which the original versions are similar) are fed into a well-trained neural network, their internal propagations have noticeable differences, which is not expected. Targeting on this problem, in this chapter, we develop the (stacked) non-local auto-encoder which exploits self-similarity information in natural images for enhancing the stability of signal propagation in the network. The self similarity property of natural images has been widely exploited in image restoration methods, which inspires the well-known non-local idea. To integrate this idea into the design of neural networks, we propose to constrain the difference between the hidden representations of non-local similar image blocks during training. Furthermore, we develop a "collaborative stabilisation" step to further rectify forward propagation. Extensive

image restoration experiments, including image denoising and super-resolution, demonstrate the effectiveness of the proposed method.

3.1 Introduction

Degradation (e.g., noise, blurring, and down-sampling) occurs during image formation in many real world imaging systems. To enhance the quality of the degraded images, applying deep models has shown to be a promising way in recent years, and has shown state-of-the-art performance [16, 88]. However, these methods are mostly executed in a data-driven manner, and seldom consider the conventional findings about image properties, such as self-similarity in natural images. These properties may also play an important role in designing modern data-driven restoration methods.

Here, we are motivated by how human neurons respond to degraded images. The human brain can be considered a powerful and constantly learning machine. During learning, vast amount of information is processed in a supervised manner before a brain response is produced. The resulting machine is very complex but can handle additional data even when data are distorted, because the brain forms a continuum of possible coding schemes during learning. These schemes have strong regularities, based on which real-world scenes or objects are robustly represented in the visual cortex in a disturbance-resistant manner. In this way, humans accomplish high-level visual tasks such as identifying objects in noisy images. A critical observation is that, due to the existence of regularities, similar visual information stimulates the brain to produce similar responses rather than randomize the transmitted brain signal [28]: the latent visual input structures need to be correctly captured by neurons without being influenced by external corruptions. This neurological observation implies that human brain possesses a stable signal propagation process, which motivates the current work.

Artificial neural networks, by definition, simulate the human central nervous system. A deep neural network can reveal abstract and distributed population structures by leveraging extensive data from the corresponding categorization. Similar to the human brain, these structures serve as the above-mentioned regularities, and in doing so help accomplish the ultimate goals of classification,

recognition, and restoration. Even though deep models has high similarity to the human visual cortex and has strong learning capacity [6, 144], there may exist some issues when applying them to image quality enhancement, for example, degradation in images may affect the stability of the deep models.

Most existing deep learning studies are based on the implication that a good model should be able to extract useful and robust hidden representations. "*Useful*" indicates that the representations are able to capture essential input information and hopefully reconstruct the inputs, while "*robust*" ensures that the representations tolerate certain input distortions. The traditional bottleneck [67], sparse representation [13, 198, 199], and denoising criterion [169] strategies are designed to encourage these two rules. This implication guarantees the effectiveness and applicability of deep learning, and the initial attempts at image restoration have shown this to be correct. However, another critical issue beyond the two rules is "*stability*" which is strongly supported by neurological observations [28]. With this in mind, we believe that *a good deep neural network should not only be able to extract useful and robust representations, but also be collaboratively stable in its internal propagation when processing similar inputs*. As well as the representation-oriented properties, this definition includes two novel characteristics:

- Stability. The hidden representations produced by the network should be stable under small input variations.
- Collaboration. During forward propagation through the network, similar inputs should be able to guide each other.

We emphasize that the proposed criterion only aims to stabilize the internal network propagation rather than constraining the distribution of the hidden representations. Based on the preceding discussion, we are concerned with how to realize an optimal deep neural network.

Here we explore non-local self-similarity in natural images to accomplish stabilization. Non-locality implies that natural images exhibit extensive pattern repetition that can be exploited to improve image restoration [15, 26, 30, 40, 59, 113, 138, 179]. From the neurological viewpoint, the human brain learns a certain type of regularity by repeatedly viewing similar visual cues to eventually produce

a stable visual system; this forms the basis of learning. Hence, our motivation stems from the fact that the knowledge of non-local similarity in natural images can provide visual cues and assist the learning of regularities in neural networks. In doing so, the resulting network can produce stable internal propagation.

With the auto-encoder paradigm in mind, we propose the *non-local auto-encoder* by exploiting the advantages of non-locality. Similar patches, which are the network inputs, are collected from training images. We define the distance between hidden representations of similar inputs as *turbulence*. Explicitly, turbulence is imposed as a regularization on the training objective of the basic auto-encoder. Implicitly, minimizing turbulence causes the network to produce similar representations for similar inputs and hence stabilizes the network. In practice, applying a well-trained non-local auto-encoder involves a feed-forward pass in which the noticable instability may reduce the output quality. Therefore, we further reduce turbulence by *collaborative stabilization*, which allows the network to produce more reliable reconstructions for corrupted inputs. Extensive experimentation on typical image restoration problems, including image denoising and super-resolution, demonstrates the effectiveness of the proposed method.

Note that the non-local concept in the proposed method is different from that in [6]. This is because our method considers spatially non-local redundancy in images, whereas the latter focuses on the non-local estimator in feature spaces and is more concerned with distributed representations. Note also that our method differs from the collaborative local auto-encoder (CLA) [29], since CLA imposes a similarity constraint on the output space while the non-local auto-encoder constrains hidden representations under a different motivation. Moreover, the proposed method involves collaborative stabilization in its forward pass, which is distinct from the direct forward pass used in CLA.

3.2 Non-local auto-encoder

Before introducing the proposed non-local auto-encoder, we first discuss our motivating concept of turbulence in neural networks.

3.2.1 Turbulence of an auto-encoder

As discussed in Section 3.1, existing constraints [13, 67, 169] are useful for structured feature learning and information preserving. However, **non-linearity in an auto-encoder may cause the failure of locality preservation in the intermediate-level representation space**. In this case, similar inputs would stimulate auto-encoder neurons and produce unstable responses. For clarity, let us assume three instances $\mathbf{y}_1$, $\mathbf{y}_2$, and $\mathbf{y}_3$ with the similarity relationship $\|\mathbf{y}_1 - \mathbf{y}_2\|_p < \|\mathbf{y}_1 - \mathbf{y}_3\|_p$, and $\mathbf{h}_i$s are the corresponding hidden representations computed via Eq. (2.1). Due to the existence of the non-linear mapping $s(\cdot)$, the distance $\|\mathbf{h}_1 - \mathbf{h}_2\|_p$, termed the *turbulence*, is NOT guaranteed to be smaller than $\|\mathbf{h}_1 - \mathbf{h}_3\|_p$. As a consequence, the network is not stable and, therefore, biases against the human neural system mechanism.

The statistical properties of turbulence are explored in Fig. 3.1. Taking the image *Lena* as an example (which is corrupted by noise), eight similar patches of size 16×16 (Fig. 3.1b) are input into the well-trained stacked auto-encoders which has six layers (Fig. 3.1a)[1]. In Fig. 3.1c, 200 neurons per layer are randomly selected and their responses are plotted for those patches. The turbulence is easily noted in these sub-figures, especially where neurons are activated (their responses are away from zero). For example, in the first and second layers, every neuron's response is unstable in the range [0.2,0.8], while in the third layer each activated neuron fails to output consistent values with respect to different patches. Conversely, for inactivated neurons in all layers, the responses are relatively stable and close to zero. We also notice from Fig. 3.1c that, as the signal goes deeper, the turbulence becomes more significant. This indicates that the hidden representations in deep layers are more sensitive to input variation than those in shallow layers. To verify this, we randomly collected 1000 sets of similar patches. For each set, the average turbulence of activated neurons in each layer was calculated; the mean and the standard deviation of the turbulence per layer are shown in Fig. 3.2. This clearly shows that the turbulence increases as the encoding signal goes deeper (see the first three layers), even though the decoding signal become relatively stable (see the last two layers).

[1]The model is built upon image patches for the consideration of their low dimensionality.

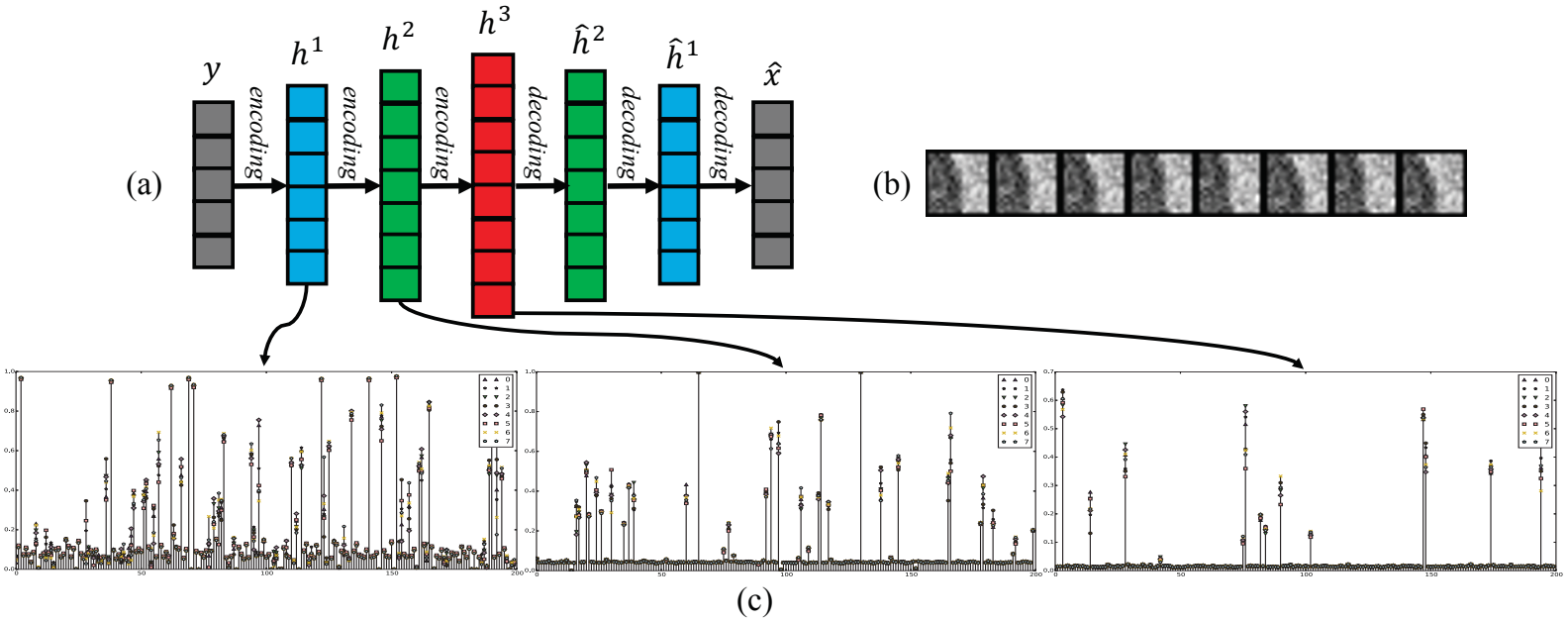


Figure 3.1: A stacked auto-encoders (in (a)) and its layer-wise responses (in (c)) with respect to the eight similar patches (in (b)). In each subfigure of (c), the x-axis indicates the indexes of neurons in the corresponding hidden layer, while the y-axis indicates the output values given those patches.

Turbulence causes problems for internal propagation in a network. Even though the network outputs can be improved by optimizing a pre-defined objective function, a biased forward propagation would reduce the final output quality. This motivates us to suppress turbulence in the proposed non-local auto-encoder.

3.2.2 Modelling the non-local auto-encoder

The proposed non-local auto-encoder architecture is very similar to the basic auto-encoder defined in Eqs. (2.1) and (2.2). However, the training objective and forward passing (in the testing phase) are different. The proposed model is illustrated in Fig. 3.3.

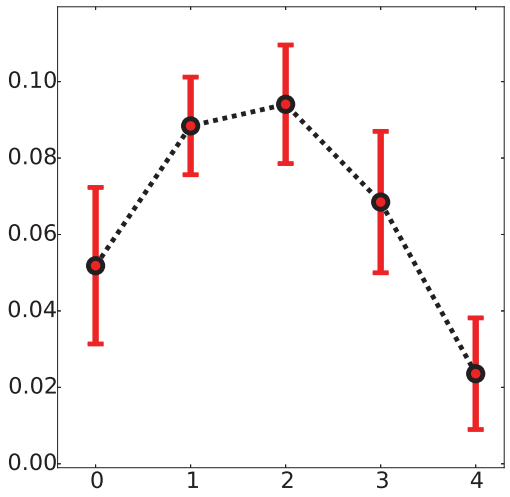


Figure 3.2: Statistics of layer-wise turbulences in stacked auto-encoders. For one of the five hidden layers, we denote by h_j the maximal response of the j-th neuron w.r.t. a set of similar patches, and by h_{max} the maximal response over all neurons in this layer. We identify neurons as activated if $h_j > 0.1 \times h_{max}$. Assuming the set of activated neurons to be $\{act\}$ with cardinality n and $p = 2$, the average turbulence is $\frac{\sqrt{\sum_{j\in\{act\}}(h_{0j}^2 - h_{1j}^2)}}{n}$, where the subscripts 0 and 1 indicate different instances. The standard derivation is calculated accordingly.

The previous discussion highlights that turbulence must be minimized during training to stabilize the resulting network. This yields a very simple regularization

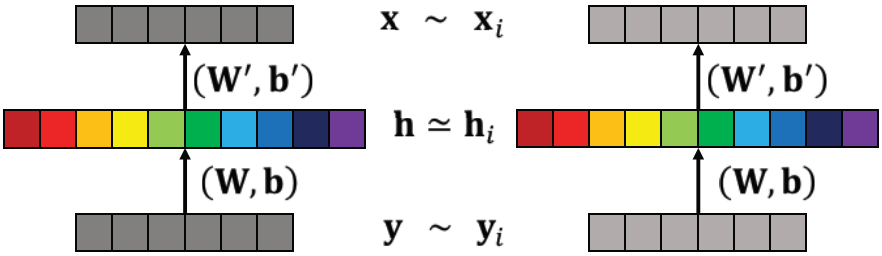


Figure 3.3: Non-local auto-encoder. $\sim$ indicates that the two arguments are similar, while $\simeq$ means that $\mathbf{h}$ and $\mathbf{h}_i$ are collaboratively similar.

term that is added to the training objective, as in most non-local-based methods. For a target patch $\mathbf{y}$, a set of similar patches $\{\mathbf{y}_i|\mathbf{y}_i \sim \mathbf{y}\}$ across the whole image are collected, thereby producing non-locality. Each input patch $\mathbf{y}_i$ is fed into the network to produce the corresponding hidden representation $\mathbf{h}_i$ via Eq. (2.1). Therefore, $\mathbf{h}_i$s are expected to be similar to $\mathbf{h}$, the hidden representation of $\mathbf{y}$. According to the previous definition, the total turbulence of all similar patches with respect to $\mathbf{y}$ can be written as

$$\sum_i \|\mathbf{h} - \mathbf{h}_i\|_p. \tag{3.1}$$

Since each $\mathbf{y}_i$ has a distinct similarity to $\mathbf{y}$, the per turbulence (i.e., $\|\mathbf{h} - \mathbf{h}_i\|_p$) should contribute differently to the above summation. Therefore, Eq. (3.1) is modified to

$$\sum_i \omega_i \|\mathbf{h} - \mathbf{h}_i\|_p, \tag{3.2}$$

where $\omega_i > 0$ is the weight assigned to the i-th instance.

Note that Eq. (3.2) is the well-known non-local regularizer when $p = 2$. The conventional non-local-based methods such as non-local means (NLM) [15] and the non-locally centralized sparse representation (NCSR) [40] assume the target pattern to be a weighted sum of its similar partners, i.e., $\mathbf{y} = \sum_i \omega_i \mathbf{y}_i$ and $\sum_i \omega_i = 1$. In this case, the objective is equivalent to $\|\mathbf{y} - \sum_i \omega_i \mathbf{y}_i\|_2 = 0$. However, this formulation is not meaningful here: because of the non-linear mapping $s(\cdot)$, the geometry of the space of hidden representations is altered. Therefore, the linear combination $\sum_i \omega_i \mathbf{h}_i$ becomes an inaccurate estimation of

$\mathbf{h}$. The following inequality clearly verifies this:

$$\begin{aligned}\|\mathbf{h} - \textstyle\sum_i \omega_i \mathbf{h}_i\|_2 &= \|s(\mathbf{W}\mathbf{y} + \mathbf{b}) - \textstyle\sum_i \omega_i s(\mathbf{W}\mathbf{y}_i + \mathbf{b})\|_2 \\ &= \|s(\mathbf{W}\textstyle\sum_i \omega_i \mathbf{y}_i + \mathbf{b}) - \textstyle\sum_i \omega_i s(\mathbf{W}\mathbf{y}_i + \mathbf{b})\|_2 \neq 0. \end{aligned} \tag{3.3}$$

This inequality is also valid when $p \neq 2$. Even though it is possible to roughly force $\mathbf{h} = \sum_i \omega_i \mathbf{h}_i$ during training, this may lead to conflict in the gradient descent of the training objective and a poorly-performing network. Therefore, we set $\omega_i = \exp(-\|\mathbf{y} - \mathbf{y}_i\|_2^2/\delta^2)$ without any normalization and directly apply Eq. (3.2) into the training objective of the non-local auto-encoder. δ is set manually as in [15].

The choice of the distance measure in Eqs. (3.1) and (3.2) is an issue. It is possible to use alternative measurements to the ℓ_p loss to control turbulence. For example, KL divergence can be used when the hidden representations have a probabilistic interpretation [67]. Meanwhile, the cosine similarity measure is also an option for non-sparse hidden representations; indeed, all these choices may be good for network stabilization. However, an implicit assumption of the ℓ_p loss attracts our attention: $\min \sum_i \|\mathbf{h} - \mathbf{h}_i\|_p$ implies that the hidden representation is corrupted by additive noise. This is important when designing the collaborative stabilization step, which is discussed in Section 3.2.4.

Combining the non-local regularizer and Eq. (2.3), we obtain the objective function of the non-local auto-encoder:

$$\ell(\theta; \mathbf{x}, \mathbf{y}) = \|\mathbf{x} - f_\theta(\mathbf{y})\|_2^2 + \lambda \sum_i \omega_i \|\mathbf{h} - \mathbf{h}_i\|_p, \tag{3.4}$$

where λ is the regularization parameter controlling the suppression of turbulence in the resulting model. In solving the above objective, the two terms balance to avoid learning an identity mapping (favoring the first term) or a zero mapping (favoring the second term). To quickly remove noisy effects in the randomly initialized $\mathbf{W}$ and $\mathbf{W}'$, we also employ the ℓ_1 regularization. Moreover, as in [16, 180], a sparse constraint is also imposed. Therefore, by integrating these

options, the whole training objective of the non-local auto-encoder becomes

$$\begin{aligned}\ell(\theta;\mathbf{x},\mathbf{y}) &= \|\mathbf{x}-f_\theta(\mathbf{y})\|_2^2+\lambda\sum_i\omega_i\|\mathbf{h}-\mathbf{h}_i\|_p \\ &\quad +\eta(\|\mathbf{W}\|_1+\|\mathbf{W}'\|_1)+\gamma KL(\rho\|\hat{\boldsymbol{\rho}}),\end{aligned} \tag{3.5}$$

where η and γ are the regularization parameters trading off the respective terms. The sparsity is controlled by the KL divergence [118, 180] in the last term, where $\hat{\boldsymbol{\rho}}=\text{avg}(\mathbf{h})$ is the average of the hidden representations of all training data. By enforcing $\hat{\boldsymbol{\rho}}$ close to a (low) fixed level ρ ($\rho=0.1$ in our experiments), the optimization will encourage hidden neuron responses to be sparse. In fact, we find from our practical experiences that the ℓ_1 regularization and the sparse regularization are very helpful in structuring the parameters $\mathbf{W}$ and $\mathbf{W}'$ when training auto-encoders. Without these two terms, optimizing either Eq. (2.3) or Eq. (3.4) often results in noisy parameters which are not expected.

3.2.3 Stacked non-local auto-encoders

Stacking the proposed models to build deep networks is similar to stacking other network units such as restricted Boltzmann machines (RBMs) in deep belief networks [66] and the basic auto-encoders [7]. Let us specify the hidden representation of the l-th layer as $\mathbf{h}^l$. The (l+1)-th non-local auto-encoder to be stacked on $\mathbf{h}^l$ is then formulated as

$$\begin{cases}\mathbf{h}^{l+1}=s(\mathbf{W}^{l+1}\mathbf{h}^l+\mathbf{b}^{l+1}),\\ \hat{\mathbf{h}}^l=s(\mathbf{W}'^{l+1}\mathbf{h}^{l+1}+\mathbf{b}'^{l+1}).\end{cases} \tag{3.6}$$

The stacked model may have different architectures in different applications. For image reconstruction here, the model contains the encoding pass and the decoding pass; for feature extraction, the model only needs the encoders, on top of which a classical classifier (e.g., support vector machine) can be used to supervise the parameter learning.

To stabilize the whole stacked network, the non-local constraint in Eq. (3.2) is imposed on the hidden representations in each intermediate layer. This can be

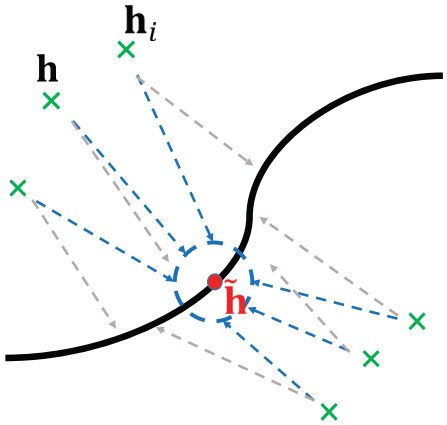


Figure 3.4: Manifold interpretation of the non-local auto-encoder.

written as $\sum_i \omega_i \|\mathbf{h}^{l+1} - \mathbf{h}_i^{l+1}\|_p$. The weights assigned to the turbulence of each instance are kept constant for all layers. To reduce the complexity of the training objective, the non-local constraints are only enforced in the encoding layers. Since layer-wise pre-training is utilized to initialize the model parameters, the similarity relationships in each of the encoding layers can be preserved in the decoding layers. Precisely, we assume a stacked model with 5 layers $\{\mathbf{y}, \mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3, \hat{\mathbf{x}}\}$. The first step of pre-training is to optimize the sub-model $\{\mathbf{y}, \mathbf{h}^1, \hat{\mathbf{x}}\}$ by using the non-local auto-encoder such that the representation $\mathbf{h}^1$ is stabilized. Then the second step is to train the sub-model $\{\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3\}$ by setting $\mathbf{h}^3 = \mathbf{h}^1$. In such a way, the stability of $\mathbf{h}^3$ (which is a decoding layer) remains the same as that of $\mathbf{h}^1$ (which is an encoding layer). Furthermore, according to the results of [46] which suggest that pre-training acts as a regularization on finetuning, the pre-trained parameters of the stacked model will be optimized in the space where the stability is assured.

3.2.4 Collaborative stabilization

During the training of the non-local auto-encoder, the parameters are directed to the space in which the resulting turbulence is most suppressed. However, given a well-trained model, the turbulence cannot be completely removed due to the limitation of local optima in the parameter space. A global optimum is extremely difficult to find in this non-convex problem. On the other hand, even

though the reconstruction error can be minimized during training, the remaining turbulence can still degenerate the forward passing process and lead to suboptimal reconstruction. To address this issue, we propose a *collaborative stabilization* step that takes advantage of multiple similar inputs. Note that this step is only deployed during testing rather than in forward propagation during training.

Recall that the hidden representation turbulence is modeled as the ℓ_p-norm of an additive noise. By taking simple derivations, minimizing Eq. (3.4) can be reformulated as maximizing the following probability:

$$P_\theta = \frac{1}{A} e^{-\|\mathbf{x}-f_\theta(\mathbf{y})\|_2^2/a^2} \prod_i \frac{1}{B_i} e^{-\|\mathbf{h}-\mathbf{h}_i\|_p/b_i^2}, \tag{3.7}$$

where a and b are hyper-parameters, A and B are the normalization parameters that ensure the unit summation of a probability. In the above equation, the reconstruction is modeled as $P(\mathbf{x}|f_\theta(\mathbf{y})) = \mathcal{N}(f_\theta(\mathbf{y}), a)$, which has a Gaussian interpretation. Since $\mathbf{h}$ and $\mathbf{h}_i$ are induced via the parametric function in Eq. (2.1), it is unsuitable to regard the second term as a simple distribution, for example Laplacian ($p = 1$) or Gaussian distribution ($p = 2$). Despite this, we find that the ℓ_p distance is to measure the significance of additive noise in the hidden representations. The hidden representations can be reasonably modeled as $\mathbf{h} = \tilde{\mathbf{h}} + \mathbf{e}$ and $\mathbf{h}_i = \tilde{\mathbf{h}} + \mathbf{e}_i$, where $\tilde{\mathbf{h}}$ is the latent true representation and $\mathbf{e}$ and $\mathbf{e}_i$ are the latent noise. Indeed, a manifold assumption [23, 169] informs us that $\tilde{\mathbf{h}}$ lies in a non-linear manifold[1]. Network instability will move $\tilde{\mathbf{h}}$ away from the manifold, resulting in the noisy $\mathbf{h}$ or $\mathbf{h}_i$, as illustrated in Fig. 3.4. The objective of training is to optimize the network parameters such that the resulting hidden representations are as close to the true ones as possible. This means that $\mathbf{h}$ is dragged from a distant to a close position (under the assumption of minimal ℓ_2 distance, for instance, the close position will be on a circle around $\tilde{\mathbf{h}}$), as indicated by the blue dashed line Fig. 3.4. The next step is to locally correct the $\tilde{\mathbf{h}}$ estimation by using its multiple observations, which is accomplished by a collaborative stabilization step as follows.

[1]While the manifold assumption in [23, 169] states that high-dimensional natural data concentrates close to a non-linear low-dimensional manifold, the non-linear function (i.e., $s(\cdot)$) can transform that manifold to an alternative non-linear manifold where the hidden representations lie.

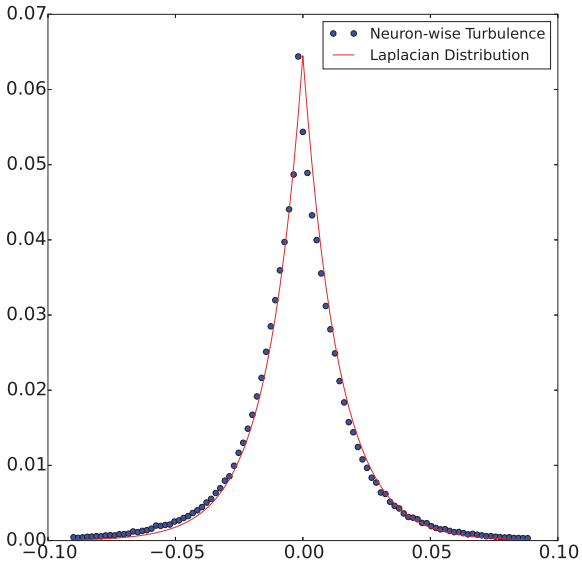


Figure 3.5: Distribution of the neuron-wise turbulence.

Given a well-trained non-local auto-encoder and a set of similar patches, $\mathbf{h}$ and $\mathbf{h}_i$s are represented as vectors. A matrix $\mathbf{D}$ is constructed by stacking these vectors column by column, such that each matrix row represents a certain neuron's responses. As discussed above, each neuron's turbulence has been significantly attenuated in training. The remaining turbulence can be viewed as additive noise imposed on the hidden representations. We also investigate the distribution of neuron-wise turbulence under a well-trained model in Fig. 3.5, which shows Laplacian characteristics. To remove the remaining turbulence, we apply adaptive Wiener filtering [94] along the rows of $\mathbf{D}$. More precisely, by denoting the j-th row of $\mathbf{D}$ as $\mathbf{d}_j$, the processor has the form

$$\mathbf{d}_j^* = \bar{\mathbf{d}}_j + \frac{\sigma_{\mathbf{d}}^2}{\sigma_{\mathbf{d}}^2 + \sigma_{\mathbf{e}}^2}(\mathbf{d}_j - \bar{\mathbf{d}}_j), \tag{3.8}$$

where $\bar{\mathbf{d}}_j$ and $\sigma_{\mathbf{d}}^2$ are the locally estimated mean and variance of signal $\mathbf{d}_j$, respectively. The variance of noise $\sigma_{\mathbf{e}}^2$ is estimated from the responses of the j-th neuron on the testing data. Once the filtering is completed, the final hidden rep-

resentations $\mathbf{h}^*$ and $\mathbf{h}_i^*$ are obtained from the resulting matrix $\mathbf{D}^*$. The network outputs are then calculated by feeding $\mathbf{h}^*$ and $\mathbf{h}_i^*$ into the decoding function Eq. (2.2).

While the above descriptions concern a single non-local auto-encoder, applying these operations to a stacked model is straightforward, with the wiener filtering simply applied layer by layer. When the hidden representations in the l-th layer are stabilized, they are then used to compute the representations in the $(l+1)$-th layer. This process is repeated until the forward propagation is completed. In this way, the remaining turbulence can be further suppressed and, therefore, the whole network is stabilized. The non-local constraint takes effect in two steps: the first step is to reduce the turbulence by optimizing the parameters, while the second step is via collaborative stabilization.

3.3 Relationships between non-local auto-encoder and the literature

Here we briefly discuss the relationships between the proposed non-local auto-encoder and other related neural networks and image restoration methods.

The non-local auto-encoder is based on previous fully-connected neural networks such as the auto-encoder [68], RBM [67], multilayer perceptron (MLP) [73], and their variants [13, 132, 169]. These are proven effective for constructing complex deep neural networks and perform well in advanced vision tasks. The primary intention of these works is to: 1) increase network capacity to represent complicated data distributions; 2) facilitate deep neural network training (via contrastive divergence and greedy layer-wise pre-training); or 3) discover latent features that are robust to data corruption (e.g., noise). In contrast, the current work is inspired by neurologic observations and the advantages of non-locality. The neurological observation informs us that the human brain should possess a stable response mechanism (see Section 3.1), and, therefore, our primary interest is in investigating how to stabilize signal propagation in a neural network. Non-local statistics reveals redundant and regular information in natural images, and is useful for the present study. Specifically, we focus on how non-local similarity

can be exploited by a deep model during both learning and testing.

We also exploit the relationship between the proposed non-local regularizer in Eq. (3.2) and some existing regularization techniques used in auto-encoders. For example, while the denoising auto-encoder [169] aims to find useful representations, the non-local auto-encoder is to ensure stable internal propagation of the network. Our method has a similar merit to the contractive auto-encoder [132] and the saturating auto-encoder [57]. These two methods constrain the distribution of hidden representations to be contractive by minimizing the Frobenius norm of Jacobian matrix [132] or by driving the hidden representations to a flat spot area of the representation space [57]. From this perspective, the proposed non-local regularizer can concentrate the hidden representations of similar inputs in space, and thus achieves the contractive purpose.

With respect to image restoration, we have already reviewed representative approaches that utilize deep learning (see Section 2) and know that some of these studies are elaborated to train deep neural networks for specific image restoration tasks [16, 60, 180, 183], and some are to design novel deep architectures [1, 35, 80] or novel data-employing policies [17]. While our work can be regarded as an extension of these studies (from the perspective of deep learning-based image restoration), the motivations are rather different. The previous approaches pay more attention to the quality of the restored images and are simply built upon basic units such as auto-encoders and CNNs. However, the proposed non-local auto-encoder differs by focusing on stabilizing internal propagation, not simply discovering useful features. The stabilization steps can remove turbulence and improve restoration performance (see the experimental results in Section 3.5). In addition, given a well-trained deep model, all previous methods estimate the restored images using a simple feed-forward pass from the bottom to the top of the model. Our method, on the other hand, employs a collaborative stabilization step to improve the signal propagation. Although the approach of Cui et al. [29] was similar (see Section 3.1), we emphasize that their method does not alter the internal propagation of the auto-encoders, and the stacked models are different.

A number of conventional image restoration methods have also partly inspired the current work. Non-locality has been thoroughly investigated under different frameworks including sparse coding and kernel regression. Since Buades et al. [15]

presented the idea of NLM, the redundancy caused by patch recurrence in natural images has become a dominant concept in modern image restoration techniques. Three approaches in particular are relevant to our method. First, Mairal et al. [113] used the non-local sparse model to force similar image patches to admit similar sparse decompositions, thereby suppressing instability in the resulting representations, similar to here. However, their work focused on the sparse coding machine rather than neural networks. Second, Dong et al. [40] used a similar but more effective approach than the first, advocating that sparse representations are corrupted by the so-called sparse coding noise, and the true representations can be estimated in a NLM manner. Although we recognize that the sparse coding noise is similar to the turbulence defined in our model, the methods used to eliminate noise and turbulence are very different. Third, the collaborative stabilization step proposed is inspired by BM3D [30], which applies a 3D linear transform (e.g., wavelet and discrete cosine transform) prior to denoising. The non-local auto-encoder, however, induces hidden representations (which are partially denoised) via a learned non-linear mapping. In this sense, our method can be viewed as a linear to non-linear and self-driven to extra data-driven extension of BM3D.

3.4 Realizations in training and testing

As indicated in [16, 80, 180], high-capacity deep neural networks are necessary to achieve acceptable image restoration performance. Training such models on large numbers of training samples is time-consuming; on the other hand, good model performance is difficult to achieve. Training is sometimes considered more of an art than a science, and in our experience different experimental settings can produce poor results without explanation. Although previous studies suggest various possible solutions to alleviate the problem, a careful design of strategies is necessary to improve training. We therefore detail the strategies employed during training and testing of the stacked non-local auto-encoders.

3.4.1 Training strategies

1) Collect similar patches in the pre-restored image:

Image corruptions can bias the similarity measurements to produce plausibly similar patches (where "plausibly" means that the uncorrupted patch version may be dissimilar). Such data conflict with our motivation, since we do not expect to project dissimilar patches into the same local space of hidden representations. Therefore, before searching for the truly similar patches, we pre-restore the corrupted images using computationally inexpensive methods. Specifically, in image denoising, NLM [15] is conducted to suppress the noise; and in image super-resolution, the low-resolution image is upscaled to the expected resolution using bicubic interpolation. Note that collecting similar patches from the uncorrupted training images is possible. However, considering that only are the degraded images observable during testing, we expect the trained model to be tolerant to an inaccurate collection of similar patches. Therefore, using the pre-restored images is reasonable for this aim. In our trials, we found that the restoration performance by using the pre-restored images in similarity search is very close to that by using the uncorrupted training images.

2) Obtain more training data:

In our experiments, we use the Berkeley segmentation dataset [116][1], which contains 500 images. Each image is degraded by the corresponding type of corruption. We extract the overlapping patches of size 16×16 from the degraded image, but perform nearest neighbors searching [8] for each patch in the pre-restored image. The number of neighbors is set to 5. In this way, about 1.4 million samples are generated, each of which is a set of 5 similar patches. Note that the inputs of our model are the corrupted patches rather than the pre-restored patches.

3) Basic architecture specifications:

We build the deep model by stacking three basic non-local auto-encoders with the hidden layer dimensions of 2048, 2048 and 2048, respectively. The model input dimension is 256. Hence, the stacked non-local auto-encoders' architecture is 256-2048-2048-2048-2048-2048-256, and in total consists of 6 layers (3 for encoding and 3 for decoding). Since a well-performing network's architecture is highly correlated with the task to be addressed, we emphasize that our defined architecture may not be optimal for all experiments.

The activation function $s(\cdot)$ is realized by the sigmoid function. We follow the

[1]https://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/

suggestion made in [16,169] that the network decoding process does NOT involve the squashing non-linearity.

4) Perform layer-wise pre-training:

Layer-wise pre-training is used to initialize deep neural network parameters. Our experience indicates that without pre-training, the whole network fine-tuning based on a set of randomly initialized parameters is very difficult to produce good parameters.

5) Set different learning rates for parameters in different layers:

In Eqs. (3.4) and (3.5), the encoder parameters (i.e., $\mathbf{W}$ and $\mathbf{b}$) contribute more than those in the decoder (i.e., $\mathbf{W}'$ and $\mathbf{b}'$) to the overall objective function (because the non-local regularizer and the sparse regularizer only affect the encoder parameters). Hence, an identical learning rate for all parameters will result in over-adjusting the lower layers and under-adjusting the upper layers. The same issue also occurs in a stacked model. To avoid this, we set the learning rates as follows. Assuming a basic learning rate of δ, the decoders are updated at the same rate δ. The learning rate of the l-th encoder (ordered from bottom to top) has an empirical setting of $\frac{1.5}{1+L-l} \times \delta$, where L is the total number of encoders ($L = 2$ in the above architecture).

6) Gradually reduce the values of the regularization parameters to lower the reconstruction error:

In experiments, we observe that as training progresses, the regularizations in Eq. (3.5) start to hinder the decrease in reconstruction error. Therefore, we reduce the regularization parameter values to gradually diminish the influence of the corresponding regularization terms. Specifically, we decrease the parameters by a factor of 10 in every 300 epochs. Note that we apply this strategy to η and γ but not to λ because the assumption of similar hidden representations must hold in the resulting model.

7) Ensure a long training time:

We set the maximal epoch to 600 in pre-training and to 4000 in fine-tuning (although the convergence is generally reached after 2000 epochs).

We adopt the adaptive gradient algorithm (ADAGRAD) [44] to optimize the whole neural network. For clarity, the key parameter settings for training are summarized in Table 3.1, which are identical in all trials. Note that these settings

Table 3.1: Parameter settings in training

Number of training samples	1,400,000
Number of similar patches in each set	5
Architecture of the stacked non-local auto-encoder	256-2048-2048-2048-2048-2048-256
ρ (expected sparsity)	0.1
λ (regularization parameter for the non-local term)	0.01
Initial η (regularization parameter for ℓ_1 term)	0.0001
Initial γ (regularization parameter for the sparse term)	0.1
Optimization algorithm	ADAGRAD
δ (learning rate)	0.01
Batch size	100
Epochs in pre-training	600
Epochs in fine-tuning	4000

are highly dependent on the training data, and the training process requires care at all times.

3.4.2 Testing strategies

We denote the degraded image as $\mathbf{Y}$, and the forward propagation of the model as $F_{cs}(\mathbf{y})$, where the subscript cs indicates that the collaborative stabilization step is utilized. The restored image is then formed as $\hat{\mathbf{X}} = \sum_{\mathbf{y}\in\mathbf{Y}} F_{cs}(\mathbf{y})$, where all estimated patch $F_{cs}(\mathbf{y})$s are aggregated. For reliable testing, we adopt simple strategies for all types of image restoration:

1) As in training, the collection of similar patches is again performed on pre-restored images. Note that pre-restoration is only for similarity estimation, but the network uses the corrupted patches as inputs.

2) For each target patch, nine similar partners are collected.

3) Since the collaborative stabilization is conducted on the hidden representations of multiple similar patches, we parallelize the reconstruction of these patches.

4) The aggregation of the estimated patches is realized by simply averaging the overlapping pixels.

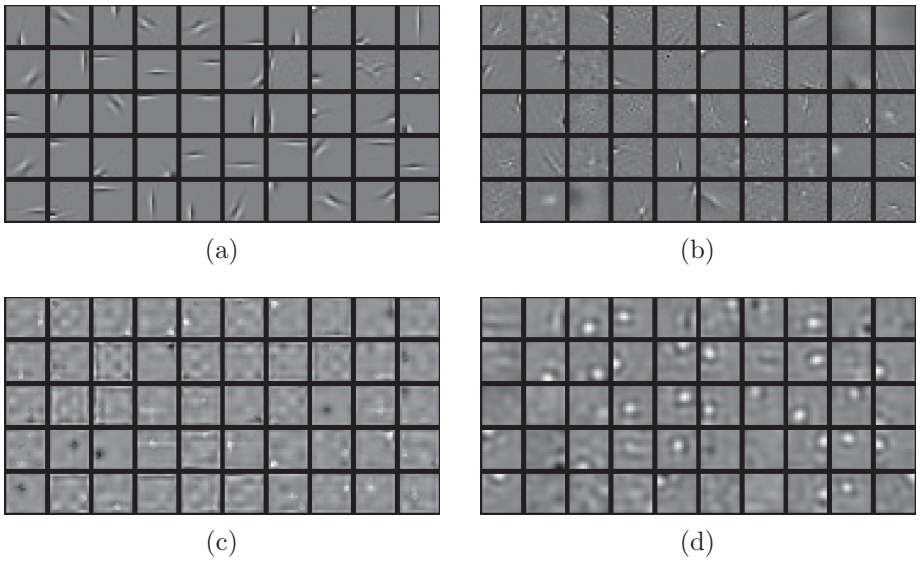


Figure 3.6: Randomly selected weights of the stacked non-local auto-encoders. (a) and (b) are from the first and last layers in image denoising task. (c) and (d) are from the first and last layers in image super-resolution task.

3.5 Experiments

In this section, we conducted extensive image denoising and super-resolution experiments based on the specifications detailed in Section 3.4. The peak signal to noise ratio (PSNR) and the structural similarity (SSIM) are used as objective assessment metrics. Meanwhile, exemplary images restored using different methods are shown in subjective assessments.

Before comparing our results with other methods, we first analyze the learned weights and stability of the well-trained stacked non-local auto-encoders in different tasks.

3.5.1 Learned weights and model stability

Since we are working on image patches, the learned weights that are applied to the input layer and that calculate the output layer can be visualized as 2D grids. In some cases, these grids are an indicator of whether the model is well trained. They also reveal what type of structure is captured or what functions are

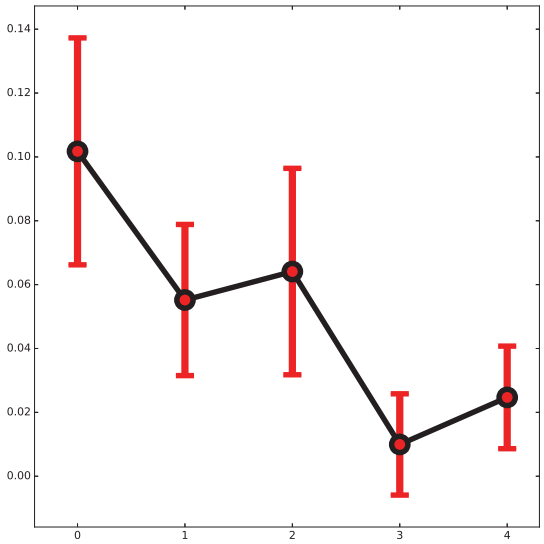


Figure 3.7: Statistics of layer-wise turbulences in stacked non-local auto-encoders.

learned. Random selections of the learned weights in image denoising and super-resolution tasks are shown in Fig. 3.6. In the denoising task, the grids in the input layer exhibit either Gabor or dot shapes. These parameters are expected to extract edges and locally salient structures from the noisy input. Similarly, the output layer grids seem to reconstruct the output by combining multiple locally concentrated structures. These plotted weights may be somewhat inconsistent with those reported by Bruger et al., in which noisy weights were observed in the solution (see [16] for further details). Indeed, in deep learning, there are probably many local solutions that are effective in the task of interest. We should point out that these weights are similar to those in the denoising auto-encoder [169], and both favor more structures over noise.

It is also necessary to note that the weights learned in super-resolution have quite different shapes to those in denoising. The input layer weights are barely noticeable except for a few dots and pillars, while most grids have a flat area. Instead, the output layer weights exhibit strange structures that are difficult to interpret. These weights might approximate an unknown interpolation function for super-resolution. However, this hypothesis requires further clarification.

To investigate model stability, we again use the image *Lena* as an example and conduct the same experiments as in Fig. 3.2. Turbulence per layer statis-

tics are shown in Fig. 3.7. Even though the first hidden layer shows relatively large turbulence, the subsequent layers exhibit reducing turbulence and increasing stability, and finally qualified outputs can be obtained. Considering that the turbulence is mainly caused by, say noise, the signal being forwarded through the network is denoised gradually. This is contradictory to the scenario seen with the stacked auto-encoders in Fig. 3.2 where the turbulences in intermediate layers are amplified. This observation may also relate to a biological viewpoint [115], which states that the mid-level visual cortex has a greater capacity to classify coherent and non-coherent image patches than the low-level visual cortex. In other words, the mid-level visual cortex should modulate more stable activity for accurate classification.

3.5.2 Image denoising

We compared the proposed stacked non-local auto-encoders (which is denoted as SNA; meanwhile, SNA_{cs} indicates that the collaborative stabilization is employed) with state-of-the-art methods including NLM [15], KSVD [45], EPLL [201], NCSR [40], BM3D [30], LSSC [113], and MLPs [16]. Among them, EPLL and MLPs are designed to exploit external image statistics, while the others basically process a single noisy image. All the testing experiments were conducted on a set of 12 natural images commonly used by the image denoising community. We know from Burger et al.'s work [16] that training a deep neural network across all noise levels may degrade model performance. Therefore, we trained the stacked non-local auto-encoders for each specific noise level, i.e. $\sigma = 25$ and $\sigma = 50$.

The detailed results of PSNR and SSIM produced by different denoising methods are shown in Tables 3.2 and 3.3. We notice that SNA is comparable to the best-performed methods, and is better than NLM, KSVD and EPLL. While the trained model has a deeper architecture than MLPs, the performance of SNA is slightly better than that of MLPs. In addition, we consider that the network with imperfect parameters is easily affected by the input corruptions. This issue can be addressed by the proposed collaborative stabilization, where the performance is further improved. We also compare the proposed SNA with a state-of-the-art method when preparing this thesis, which is DnCNN [194]. As shown, benefitting

Table 3.2: PSNR(dB) and SSIM results of image denoising ($\sigma = 25$). The best performance among the competitors (except DnCNN) is marked in bold.

Method	NLM [15]	KSVD [45]	EPLL [201]	NCSR [40]	BM3D [30]	LSSC [113]	MLPs [16]	SNA	SNA$_{cs}$	DnCNN [194]
barbara	28.09/0.7803	29.61/0.8498	28.61/0.8464	30.63/0.8856	**30.70/0.8859**	30.51/0.8809	29.54/0.8640	29.46/0.8630	29.48/0.8638	30.00/0.8778
boat	27.93/0.7131	29.30/0.7707	29.69/0.7944	29.69/0.7919	29.84/0.7990	29.86/0.7959	29.93/0.8011	29.91/0.8017	**29.93/0.8025**	30.18/0.8085
cameraman	27.75/0.7543	28.98/0.8357	29.22/0.8523	29.38/0.8501	29.50/0.8503	29.43/0.8512	29.54/0.8590	29.61/0.8600	**29.63/0.8606**	30.06/0.8694
couple	27.46/0.7153	28.90/0.7769	29.50/0.8069	29.45/0.8058	**29.70/0.8167**	29.62/0.8115	29.70/0.8151	29.62/0.8133	29.64/0.8143	30.06/0.8253
fingerprint	26.04/0.8680	27.28/0.8944	27.20/0.9029	**27.85**/0.9103	27.73/0.9094	27.64/0.9073	27.67/0.9109	27.81/0.9157	27.84/**0.9166**	27.63/0.9111
hill	28.06/0.6907	29.14/0.7369	29.54/0.7651	29.64/0.7700	29.76/**0.7741**	29.73/0.7737	29.76/0.7734	29.76/0.7729	**29.78**/0.7739	29.93/0.7791
house	30.09/0.7563	32.17/0.8475	32.19/0.8547	**33.01**/0.8620	32.89/0.8614	33.21/**0.8679**	32.47/0.8586	32.43/0.8549	32.44/0.8551	33.08/0.8651
lena	29.67/0.7498	31.37/0.8442	31.77/0.8557	31.96/0.8621	32.08/0.8614	31.90/0.8575	**32.34/0.8676**	32.24/0.8652	32.26/0.8656	32.50/0.8707
man	28.08/0.7162	29.11/0.7785	29.61/0.8041	29.54/0.8004	29.56/0.8008	29.63/0.8027	29.82/0.8113	29.85/0.8112	**29.87/0.8121**	30.04/0.8185
Monarch	27.42/0.8087	29.06/0.8897	29.56/0.9024	29.65/0.9033	29.38/0.9009	29.56/0.9002	29.74/0.9040	29.75/0.9039	**29.78/0.9044**	30.52/0.9187
peppers	28.03/0.7761	29.82/0.8600	30.24/**0.8750**	30.12/0.8708	30.21/0.8716	30.32/0.8694	30.28/0.8745	30.63/0.8710	**30.66**/0.8714	30.86/0.8849
straw	24.62/0.8149	25.78/0.8508	25.83/0.8609	26.26/0.8723	25.99/0.8647	26.25/0.8751	26.24/0.8742	26.39/0.8802	**26.42/0.8818**	26.61/0.8881
Average	27.77/0.7620	29.21/0.8279	29.41/0.8434	29.77/0.8487	29.78/0.8497	29.80/0.8494	29.75/0.8511	29.79/0.8511	**29.81/0.8518**	30.12/0.8598

Table 3.3: PSNR(dB) and SSIM results of image denoising ($\sigma = 50$). The best performance among the competitors (except DnCNN) is marked in bold.

Method	NLM [15]	KSVD [45]	EPLL [201]	NCSR [40]	BM3D [30]	LSSC [113]	MLPs [16]	SNA	SNA$_{cs}$	DnCNN [194]
barbara	23.85/0.5569	25.51/0.7142	24.91/0.7100	27.08/0.7898	**27.26/0.7949**	27.13/0.7858	25.45/0.7363	25.79/0.7520	25.80/0.7523	26.23/0.7715
boat	24.19/0.5056	25.84/0.6545	26.63/0.6964	26.47/0.6921	26.64/0.7008	26.70/0.6978	26.94/0.7100	26.96/0.7137	**26.97/0.7141**	27.11/0.7162
cameraman	24.11/0.5327	25.68/0.7386	26.11/0.7703	26.03/0.7760	26.13/0.7782	26.38/0.7788	26.40/0.7869	26.52/0.7965	**26.53/0.7967**	27.06/0.7974
couple	23.81/0.4975	25.28/0.6303	26.33/0.6925	26.23/0.6923	26.45/0.7033	26.33/0.6932	**26.73/0.7144**	26.66/0.7128	26.67/0.7133	26.92/0.7226
fingerprint	21.66/0.6927	23.23/0.7490	23.66/0.7948	24.48/0.8171	**24.52/0.8265**	24.25/0.8171	24.17/0.8180	24.34/0.8261	24.35/0.8265	24.10/0.8188
hill	24.56/0.4816	26.32/0.6264	27.02/0.6676	26.98/0.6629	27.14/0.6746	27.15/0.6713	**27.35**/0.6833	27.33/0.6839	27.34/**0.6843**	27.38/0.6851
house	25.48/0.5287	28.04/0.7659	29.10/0.8037	29.65/0.8206	29.66/0.8128	**29.86/0.8146**	29.56/0.8149	29.70/0.8181	29.71/0.8181	30.11/0.8236
lena	25.59/0.5252	27.81/0.7607	28.64/0.7896	28.96/0.8074	29.07/0.8006	28.92/0.7915	29.38/0.8112	29.38/0.8135	**29.39/0.8136**	29.41/0.8109
man	24.53/0.5021	26.11/0.6650	26.80/0.7025	26.65/0.7005	26.82/0.7060	26.74/0.6990	27.11/0.7164	27.12/0.7196	**27.13/0.7199**	27.22/0.7218
Monarch	23.55/0.6187	25.31/0.7966	25.93/0.8294	25.66/0.8248	25.78/0.8244	25.71/0.8161	26.25/0.8318	26.42/0.8390	**26.43/0.8393**	26.74/0.8482
peppers	23.86/0.5678	26.16/0.7729	26.80/0.7976	26.60/0.7983	26.72/0.7903	26.72/0.7869	26.74/0.7971	27.38/0.8029	**27.39/0.8031**	27.31/0.8099
straw	20.56/0.5510	21.52/0.5795	21.93/0.6372	22.43/0.6843	22.36/0.6828	22.60/0.7127	22.57/0.7033	22.83/0.7219	**22.84/0.7230**	22.80/0.7364
Average	23.81/0.5467	25.57/0.7045	26.15/0.7410	26.43/0.7555	26.55/0.7579	26.54/0.7554	26.55/0.7603	26.70/0.7667	**26.71/0.7670**	26.87/0.7719

Table 3.4: Effect of non-local regularizer on image denoising. Average performances are reported.

	SA	MLPs	SNA
$\sigma = 25$	29.43/0.8467	29.75/0.8511	**29.79/0.8511**
$\sigma = 50$	26.00/0.7322	26.55/0.7603	**26.70/0.7667**

from the advanced training strategy, DnCNN achieves the best performance in almost all cases. For visual comparison, Figs. 3.8 and 3.9 illustrate the examples for $\sigma = 25$ and $\sigma = 50$, respectively.

We also investigate the performances of SNA and SNA$_{cs}$ under different noise levels. Specifically, the model trained for $\sigma = 25$ was applied to restore the images corrupted by noise ranging from $\sigma = 5$ to $\sigma = 85$. The curves of average PSNR *versus* σ are plotted in Fig. 3.10. As shown, when $\sigma < 25$, the collaborative stabilization slightly decreases the performance of SNA, which is caused by the over-smoothed details. When $25 \leq \sigma \leq 55$, however, SNA$_{cs}$ is more robust than SNA to the noise variation. In the case of severe noise where $\sigma > 55$, both methods cannot restore the image details effectively.

To verify the effect of the proposed non-local regularizer on image denoising, we compare the performances of the stacked auto-encoders trained with and without the regularizer. We notice that the stacked auto-encoders (SA) is similar to MLPs but with different architectures. The results are listed in Table 3.4, which reveal that SNA outperforms both the stacked auto-encoders and MLPs.

Running time: We compare the efficiency of different methods in Table 3.5. We implement our method with Python and Theano, and use a GPU device for parallel computation. As shown, SNA has a comparable efficiency to MLPs. SNA$_{cs}$ is much slower than SNA since the collaborative stabilization process requires searching for similar patches, which is computationally expensive. DnCNN is fast owing to the acceleration by the Nvidia cuDNN-v5 deep learning library.

3.5.3 Image super-resolution

In our image super-resolution experiments, we compared SNA with four compeititve approaches: ScSR [189], NCSR [40], ANR [164], and CNN [35]. All

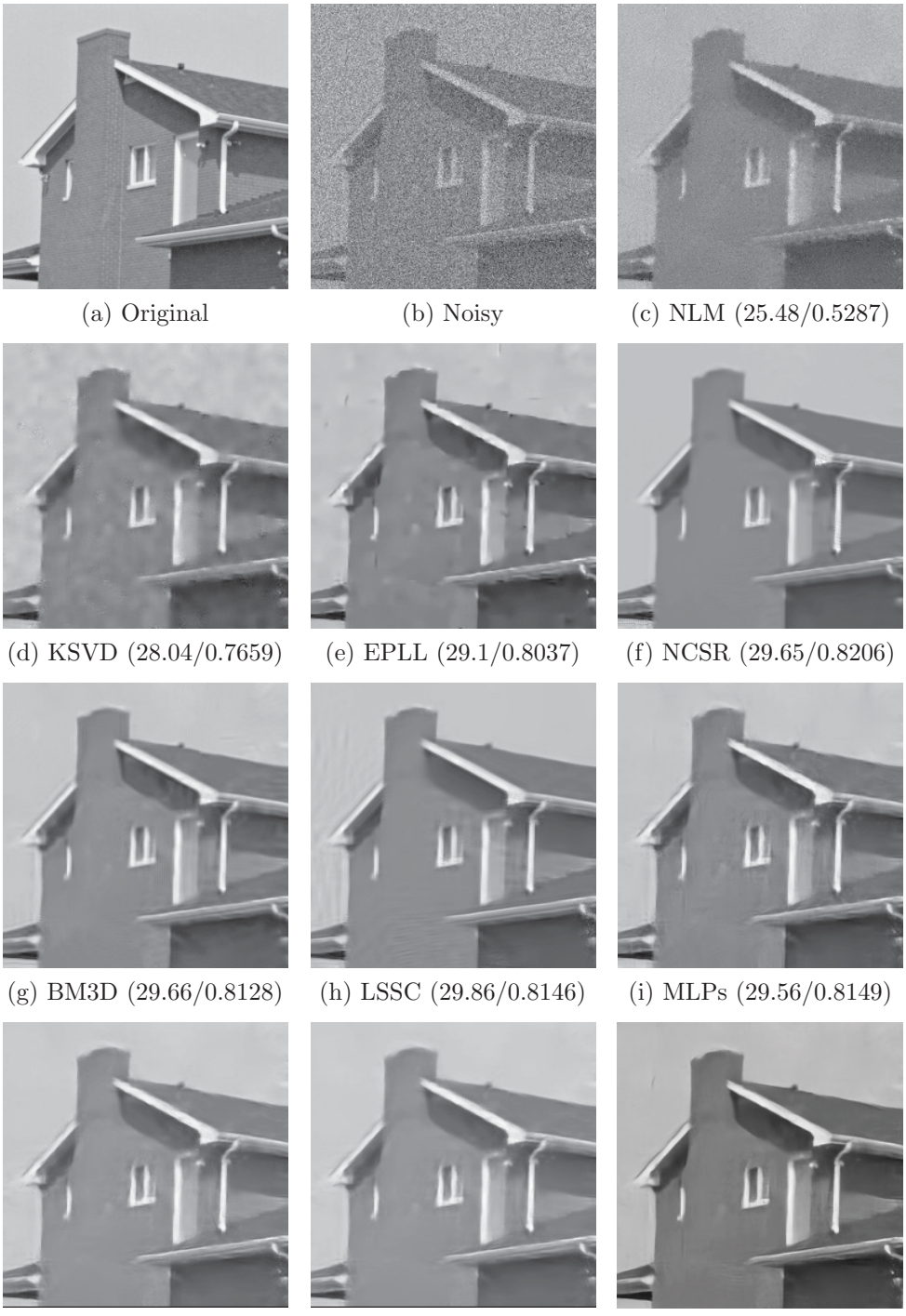
(a) Original (b) Noisy (c) NLM (29.67/0.7498)

(d) KSVD (31.37/0.8442) (e) EPLL (31.77/0.8557) (f) NCSR (31.96/0.8621)

(g) BM3D (32.08/0.8614) (h) LSSC (31.9/0.8575) (i) MLPs (32.34/0.8676)

(j) SNA (32.24/0.8652) (k) SNA_{cs} (32.26/0.8656) (l) DnCNN (32.50/0.8707)

Figure 3.8: PSNR(dB) and SSIM results of different image denoising methods on *lenna* ($\sigma = 25$).



(a) Original (b) Noisy (c) NLM (25.48/0.5287)

(d) KSVD (28.04/0.7659) (e) EPLL (29.1/0.8037) (f) NCSR (29.65/0.8206)

(g) BM3D (29.66/0.8128) (h) LSSC (29.86/0.8146) (i) MLPs (29.56/0.8149)

(j) SNA (29.70/0.8181) (k) SNA_{cs} (29.71/0.8181) (l) DnCNN (32.50/0.8707)

Figure 3.9: PSNR(dB) and SSIM results of different image denoising methods on *house* ($\sigma = 50$).

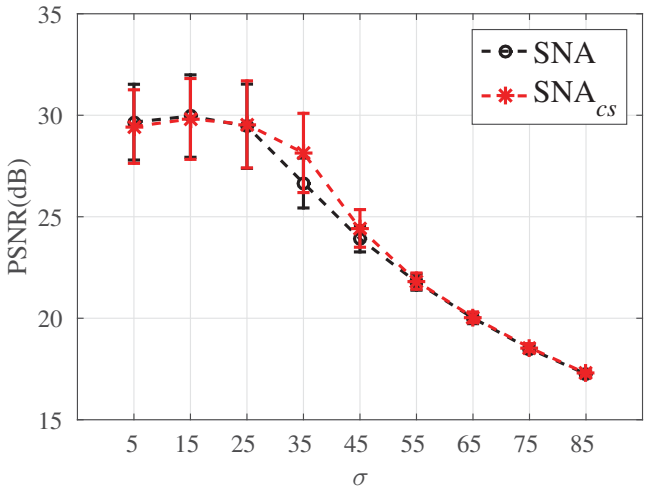


Figure 3.10: Performances of SNA and SNA$_{cs}$ (both of which are trained for $\sigma = 25$) with respect to different noise levels.

methods were tested by assigning upscaling factors of 2 and 4. Two popular image sets, Set5 [10] and Set14 [193], were selected as test data. To generate low-resolution images, we followed the common technique that employs the *bicubic* interpolation to reduce the image resolution. We trained dictionaries using the data generated via the above degradation for ScSR, NCSR and ANR. Considering that human vision is more sensitive to luminance variations, the above methods were only applied to the luminance channel while bicubic interpolation was conducted on the chromatic components.

The results of different methods are compared in Tables 3.6 and 3.7. As observed, SNA outperforms all the other competitors in terms of the average PSNR values; and for the average SSIM values, our model is again the best except for the case of the scaling factor 2 on Set5. These results are indicative that SNA can handle different scales and different images well, and therefore are superior. The lower SSIM values of SNA than CNN may be because CNN has a larger receptive field size and produce clearer textures. For subjective assessments, we display the restored images in Figs. 3.11 and 3.12. The enlarged areas indicate that SNA can produce much sharper edges than the others, and reconstruct the best visually pleasant words and eyelashes.

A low-resolution image is possible to be corrupted by noise in real applications.

Table 3.5: Running time comparison between different methods on image denoising. The methods with GPU implementations are marked with "*".

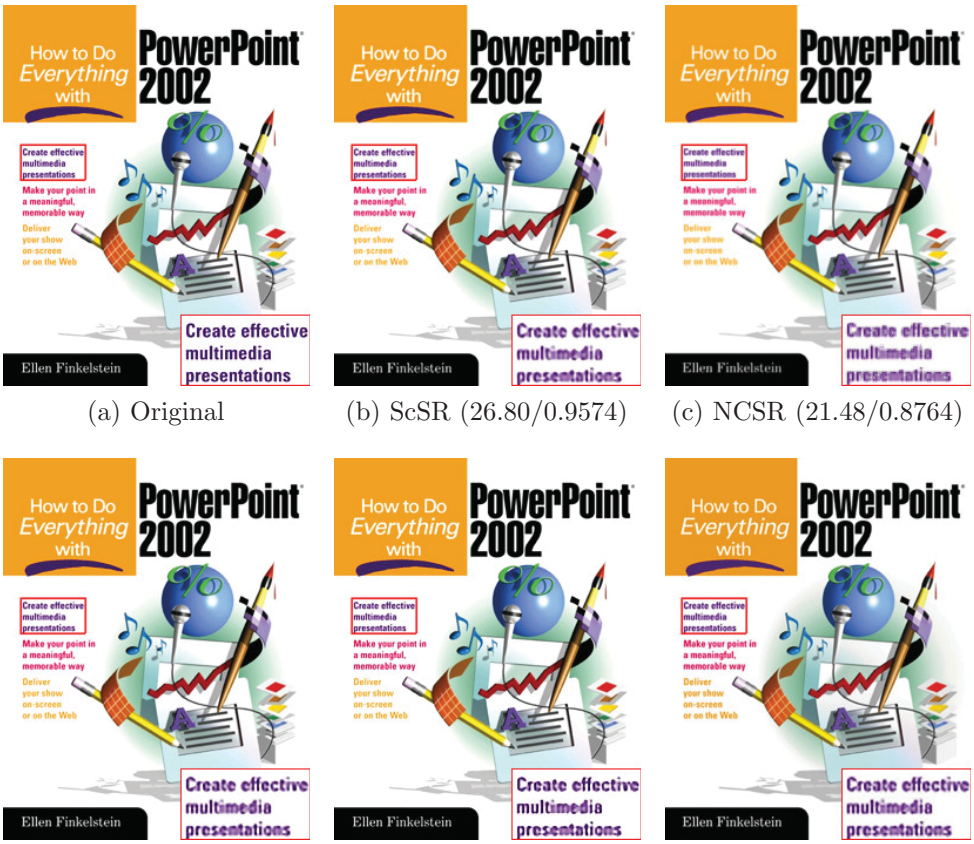
Method	Running Time (s)
NLM [15]	12.22
KSVD [45]	8.89
EPLL [201]	172.32
NCSR [40]	852.39
BM3D [30]	2.69
LSSC [113]	1041.27
MLPs [16]	7.71
DnCNN* [194]	4.16
SNA*	8.50
SNA_{cs}*	40.24

Table 3.6: PSNR(dB) and SSIM results of image super-resolution on *Set5*.

Scale	2				
Method	ScSR [189]	NCSR [40]	ANR [164]	CNN [35]	SNA
baby	35.87/0.9496	30.07/0.8861	37.16/**0.9615**	37.09/0.9611	**38.17**/0.9614
bird	36.53/0.9735	29.80/0.8775	38.79/0.9833	39.24/**0.9838**	**39.34**/0.9805
butterfly	27.97/0.9359	24.29/0.8870	29.17/0.9468	**31.55/0.9621**	30.91/0.9485
head	33.72/0.8521	29.47/0.7031	34.39/0.8682	34.13/0.8648	**35.45/0.8794**
woman	31.37/0.9495	27.23/0.8920	33.23/0.9630	33.95/**0.9655**	**34.40**/0.9623
Average	33.09/0.9321	28.17/0.8491	34.55/0.9446	35.19/**0.9474**	**35.65**/0.9464
Scale	4				
Method	ScSR [189]	NCSR [40]	ANR [164]	CNN [35]	SNA
baby	30.42/0.8356	26.02/0.7472	31.72/0.8685	31.67/0.8665	**33.21/0.8840**
bird	28.94/0.8564	24.59/0.7045	30.51/0.8896	31.03/0.8969	**32.28/0.9071**
butterfly	21.19/0.7482	19.35/0.6919	22.20/0.7676	24.36/**0.8441**	**25.13**/0.8432
head	30.28/0.7196	26.85/0.5904	30.99/0.7472	30.94/0.7441	**32.25/0.7758**
woman	25.17/0.8170	22.42/0.7368	26.48/0.8491	27.51/0.8721	**28.42/0.8765**
Average	27.20/0.7954	23.85/0.6942	28.38/0.8244	29.10/0.8447	**30.26/0.8573**

Table 3.7: PSNR(dB) and SSIM results of image super-resolution on *Set14*.

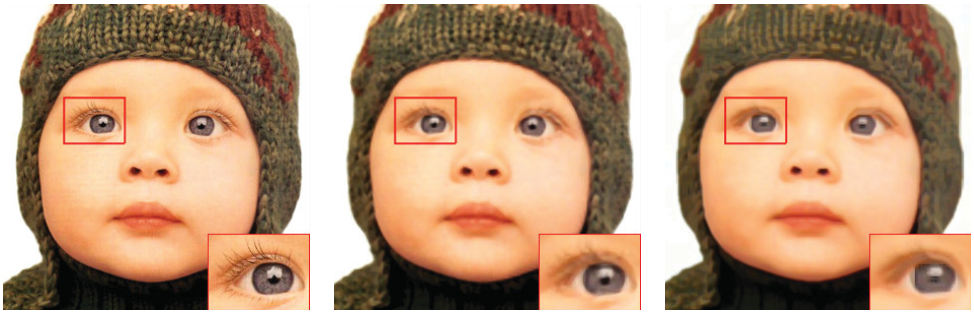
Scale	2				
Method	ScSR [189]	NCSR [40]	ANR [164]	CNN [35]	SNA
baboon	23.66/0.6986	22.43/0.5871	24.22/0.7439	24.44/**0.7566**	**25.55**/0.7561
barbara	26.70/0.8393	25.40/0.7893	27.28/0.8673	27.31/0.8705	**28.65/0.8723**
bridge	26.83/0.8083	24.84/0.6951	27.54/0.8412	**27.84/0.8500**	27.52/0.8382
coast.	28.13/0.7884	25.88/0.6641	29.12/0.8359	29.31/0.8438	**30.44/0.8479**
comic	25.35/0.8728	22.71/0.7727	26.45/0.9009	27.19/**0.9146**	**27.70**/0.8996
face	33.69/0.8519	29.48/0.7038	34.36/0.8682	34.11/0.8648	**35.42/0.8794**
flowers	29.94/0.9056	26.62/0.8300	30.97/0.9229	32.01/**0.9307**	**32.32**/0.9258
foreman	33.84/0.9501	29.13/0.9113	35.08/0.9599	34.68/**0.9610**	**35.68**/0.9588
lenna	33.64/0.8998	30.14/0.8524	35.02/0.9150	35.36/0.9175	**36.18/0.9246**
man	28.49/0.8471	25.60/0.7068	29.15/0.8695	29.60/0.8745	**30.37/0.8756**
monarch	33.14/0.9623	29.38/0.9360	34.41/0.9698	**36.54/0.9744**	35.96/0.9693
pepper	34.31/0.8933	30.25/0.8514	35.09/0.9021	35.51/0.9044	**36.12/0.9130**
ppt3	26.80/0.9574	21.48/0.8764	27.59/0.9638	29.96/**0.9715**	**30.11**/0.9710
zebra	29.96/0.9112	25.15/0.7814	31.75/0.9335	31.43/0.9339	**33.34/0.9390**
Average	29.61/0.8704	26.32/0.7827	30.57/0.8924	31.09/0.8977	**31.81/0.8979**
Scale	4				
Method	ScSR [189]	NCSR [40]	ANR [164]	CNN [35]	SNA
baboon	21.05/0.4148	20.29/0.3414	21.36/0.4690	21.41/0.4775	**22.74/0.4996**
barbara	23.72/0.6612	22.41/0.6015	24.28/0.7023	24.44/0.7153	**25.63/0.7245**
bridge	23.10/0.5289	21.69/0.4313	23.62/0.5855	**23.76/0.5957**	23.73/0.5918
coast.	24.14/0.4757	23.24/0.4211	24.47/0.5139	24.64/0.5208	**25.98/0.5573**
comic	20.37/0.5701	18.83/0.4631	21.00/0.6297	21.37/0.6574	**22.62/0.6633**
face	30.21/0.7169	26.80/0.5887	30.95/0.7453	30.87/0.7420	**32.19/0.7737**
flowers	24.28/0.7045	22.22/0.6175	25.15/0.7439	25.80/0.7628	**27.05/0.7765**
foreman	28.76/0.8519	25.36/0.7987	29.50/0.8620	30.89/0.8875	**31.69/0.8924**
lenna	28.62/0.7880	25.46/0.7229	29.67/0.8171	30.14/0.8232	**31.37/0.8449**
man	24.47/0.6440	22.29/0.5119	25.11/0.6837	25.46/0.6981	**26.73/0.7232**
monarch	26.38/0.8746	24.10/0.8253	27.39/0.8928	29.05/0.9122	**29.99/0.9166**
pepper	29.54/0.8100	25.53/0.7522	30.62/0.8281	31.48/0.8380	**32.60/0.8595**
ppt3	20.76/0.8149	18.45/0.7504	21.42/0.8203	23.40/0.8728	**24.31/0.8841**
zebra	22.82/0.6470	20.31/0.5047	24.13/0.7020	24.44/0.7160	**26.29/0.7550**
Average	24.87/0.6788	22.64/0.5951	25.62/0.7140	26.23/0.7299	**27.35/0.7473**



(a) Original (b) ScSR (26.80/0.9574) (c) NCSR (21.48/0.8764)

(d) ANR (27.59/0.9638) (e) CNN (29.96/0.9715) (f) SNA (30.11/0.9710)

Figure 3.11: Image super-resolution results of different methods when the upscale factor is 2.



(a) Original (b) ScSR (30.42/0.8356) (c) NCSR (26.02/0.7472)

(d) ANR (31.72/0.8685) (e) CNN (31.67/0.8665) (f) SNA (33.21/0.8840)

Figure 3.12: Image super-resolution results of different methods when the upscale factor is 4.

Even though the collaborative stabilization may over-smooth the fine details in the upscaled noiseless images, SNA_{cs} works better than SNA when noise exists. This is illustrated in Fig. 3.13.

The effect of the non-local regularizer on image super-resolution is also investigated. The results are detailed in Table 3.8, which suggest that the regularizer has limited impact on the noiseless super-resolution task.

The comparison of efficiency between different methods is illustrated in Table 3.9. As shown, SNA has relatively high efficiency among the other methods.

3.6 Conclusions

Deep learning has achieved state-of-the-art performance in many applications and has recently been applied to various image restoration tasks. However, most existing deep models have been developed to discover useful latent features in

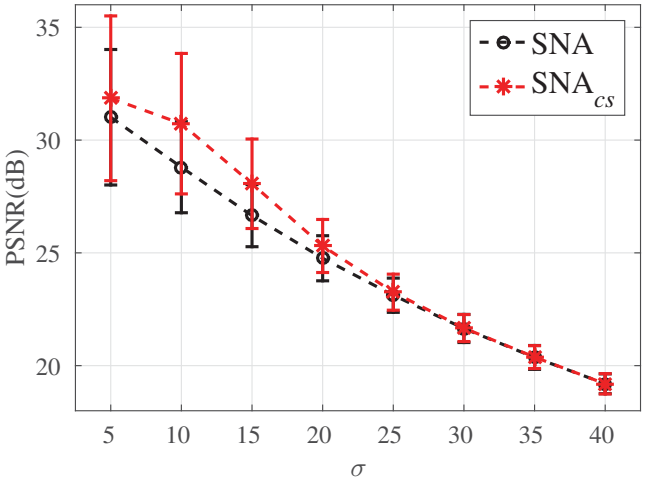


Figure 3.13: Super-resolution performances of SNA and SNA$_{cs}$ in handling noisy low-resolution images.

Table 3.8: Effect of non-local regularizer on image super-resolution. Average performances are reported.

	SA	SNA
upscale factor = 2	32.64/0.9046	**32.82/0.9107**
upscale factor = 4	28.01/0.7657	**28.12/0.7762**

a large amount of training data. We argue that a good neural network should not only extract useful information but also be stable in its internal propagation. Explicitly, similar inputs should induce similar hidden representations throughout the network. Any input data corruption could destroy such a criterion and thus degrade output quality. To address this issue, we minimized the turbulence, defined as the difference between the hidden representations of similar inputs. This motivated the development of the non-local auto-encoder, in which the turbulence was suppressed via two steps: first, by evolving a non-local regularizer in the training objective, and second, by the proposed collaborative stabilization. The non-local auto-encoder is an extension of the basic auto-encoder. Therefore, stacking and layer-wise pre-training of our model are similar. Even though the stacked deep model training algorithm could be simple and straightforward, careful design guarantees an effective model. To this end, we discussed the prac-

Table 3.9: Running time comparison between different methods on image SR with scale factor 2. The methods with GPU implementations are marked with "*".

Method	Running Time (s)
ScSR [189]	157.85
NCSR [40]	370.16
ANR [164]	1.07
CNN [35]	4.77
SNA*	5.21

tical aspects of training and testing. The effectiveness of the proposed model was demonstrated by way of extensive image denoising and super-resolution experiments. The stacked non-local auto-encoders achieved comparable denoising performance to other state-of-the-art methods and outperformed other leading super-resolution methods.

Chapter 4

Receptive Field Size *vs.* Model Depth

When designing a deep architecture for image restoration, there are many factors which should be considered carefully and which may have substantial influences on the performance, such as receptive field size, model depth, multi-scale combination, residual learning, just to name a few. These factors have been inadequately investigated in existing works from which we find a claim saying that a larger receptive field size or a deeper model can result in better performance. However, we are concerned about the question that which factor, receptive field size or model depth, is more critical. While a deeper model may have a larger receptive field size, it can also bring more computational cost. It is actually possible to increase one while fixing another, make a trade-off between the performance and the cost. Thus, we are interested in revealing the answers of that question. In this chapter, we focus on the single image super-resolution (SISR) task, and propose a strategy based on dilated convolution to investigate how the two factors affect the performance of SISR. Our findings from exhaustive investigations suggest that SISR is more sensitive to the changes of receptive field size than the variations of model depth, and that the model depth must be congruent with the receptive field size to produce improved performance. These findings inspire us to design a shallower architecture which can save computational and memory cost while preserving comparable effectiveness with respect to a much deeper architecture.

4.1 Introduction

A high-resolution (HR) image with delicate details is highly expected for either aesthetics or applications such as computer vision, medical imaging, video surveillance, and entertainment. Unfortunately, many undesirable imaging conditions hinder the acquisition of such images, but result in low-resolution (LR) substitutes. This poses the necessity of single image super-resolution (SISR), which aims at recovering a high-resolution image from its corresponding single low-resolution image. The state-of-the-art performance on this task has been achieved by deep learning-based methods [14, 88, 89, 149, 172], telling us a fact that a deeper model has higher nonlinearity and larger receptive field, which is beneficial for boosting the performance of SISR.

While the current SISR achievements are fascinating, we still have questions about the CNN architecture, which has been extensively studied in the deep learning community [3]. Does the CNN architecture for SISR really need to be deep (or need so high nonlinearity)? How large receptive field does SISR require? Which is more dominant among model depth (or model nonlinearity) and receptive field size? We interpret these questions from the perspective of SISR as: Is it beneficial to apply a complex function on a very local region to estimate the HR pixel? Or is it good to use a relatively simpler function on a wide region? Or are both a complex function and a wide region needed? These questions inspire us to deeply research and understand which aspects are critical for SISR.

To pursue the answers, we are firstly aware that a larger receptive field of a CNN model captures more contextual information. We use the term "the receptive field of a CNN model" to denote the receptive field of the output layer of the model. Most image classification networks [64, 93, 150, 161] employ successive pooling and subsampling operations to gradually reduce the resolution of feature maps and extend the size of the receptive field, such that the output layer can respond to the whole input image while maintaining a manageable memory cost. But for the SISR task, a dense prediction of all pixels in the HR image is needed, which prevents the reduction of the feature map resolution, because the pooling and subsampling operations will cause information lost. To mimic a

similar behaviour in SISR, we propose to expand the size of the receptive field by using dilated convolution, without sacrificing the resolution and stacking the CNN layers to an unnecessary depth.

Dilated filters have been widely used in the wavelet theory [72,114,148]. They are the key concept for realising the multi-resolution analysis, where the mother wavelets are scaled (or dilated) and translated according to a dilation parameter and a translation parameter. According to the lemma proposed by Holschneider *et al.* [72], the convolution with a dilated filer can be factorised as simpler convolutions. The proposed *algorithme à trous* is similar to the convolution and subsampling operations widely used in CNN. In our work, to prevent resolution reduction, we can perform an inverse factorisation, which means that the subsampling and convolution operations are replaced by a dilated convolution. Instead of explicitly representing the dilated filter based on the learned convolutional filters, the filter parameters of the dilated convolutional layer are trained in conjunction with other layers in the CNN model.

In this chapter, we propose to integrate the dilated convolutional layer into CNN models to investigate the questions raised previously. To the best of our knowledge, this is the first attempt to apply the dilated convolution in the SISR task. In specific, we investigate the effects of different receptive field sizes on SISR by keeping constant model nonlinearity. Dilated convolutional layers are optionally used to enlarge the receptive field of the CNN models. It is also studied that how an increased receptive field size affects the SISR performance of different kinds of images, given a fixed model depth. In this task, we are towards finding the effective contextual region size required by SISR. For another task, we investigate the effects of the model depth on SISR by fixing the receptive field size. The convolutional layers with 1×1 filters can be stacked to increase the nonlinearity while keeping receptive field size unchanged. We try to analyse from the results whether the nonlinearity is an important aspect for SISR. The findings reveal the dominant relationship and the interdependent relationship between the two factors, which can guide the designs of effective and efficient CNN models for SISR.

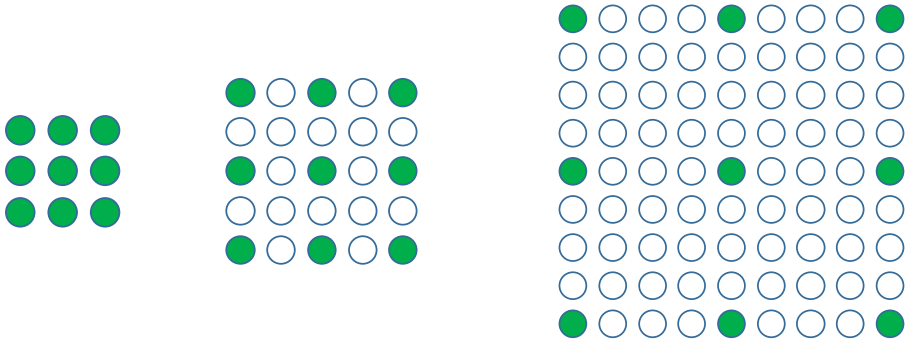


Figure 4.1: 2D dilated filters. The solid circles indicate the filter parameters, while the hollow circles indicate zeros that are inserted during dilation.

4.2 Related Work on Dilated Convolution

In the field of wavelet decomposition, a signal is decomposed through a series of convolutions with filters that have different scales. These filters form a filter bank, which can be generated by introducing a dilation factor into the so-called mother wavelet [72, 114, 168]. In mathematics, let $f(t)$ be a function in the space $L_2(\mathbb{R})$, and $\psi(t) \in L_2(\mathbb{R})$ be a filter which is dilated via $\psi_a(t) = \psi(\frac{t}{a})$. The convolutions with a non-dilated filter and with a dilated filter are, respectively, expressed as,

$$(f * \psi)(t) = \int_\tau f(\tau)\psi(t-\tau)d\tau, \tag{4.1}$$

$$(f * \psi_a)(t) = \int_\tau f(\tau)\psi(\frac{t-\tau}{a})d\tau, \tag{4.2}$$

where $*$ is the convolution operation. In discrete case, Eq. (4.2) is rewritten as

$$(f * \psi_a)[k] = \sum_\upsilon f[\upsilon]\psi[\frac{k-\upsilon}{a}]. \tag{4.3}$$

The *algorithme à trous* implements the convolution with a dilated filter through the convolution with a small filter followed by subsampling. Such an implementation indicates that, in discrete case, ψ is dilated by zeros to generate ψ_a, which

means $\psi_a[k] = 0$ when k is divisible by a. Then, Eq. 4.3 is equal to

$$(f * \psi_a)[k] = \sum_{\upsilon} f[k - a\upsilon]\psi[\upsilon], \tag{4.4}$$

which is referred as the dilated convolution operation by Yu and Koltun et al. [190]. A dilated convolutional layer can be integrated into a deep CNN model, where the filter ψ is learned via back-propagation. This is different from the dilated filters in wavelet decomposition, which must be constructed under rigorous conditions, such as orthonormality.

Suppose that ψ has a finite support of r in discrete case, then the effective size of ψ_a is calculated as $a(r-1)+1$. The value of the dilation factor a is chosen to be exponentially increased, such that the subsampling operation either in *algorithme à trous* or in the 2-stride pooling layer of deep models can be compensated, *i.e.* $a = 2^i$ for $i = 0, 1, ..., n$. The generalisation of the above formulations to 2D case is straightforward and omitted here. Fig. 4.1 gives an illustration of the 2D dilated filters.

Dilated convolutional layers have recently been employed in the image segmentation task [24, 25, 190]. The motivations for using dilation are, on one hand, to avoid reducing the resolution of CNN feature maps and increasing the number of model parameters, and on the other hand, to maximally expand the receptive field to grasp global information of the input image.

4.3 Basic Settings for Comparison

In this section, we introduce the basic necessary settings of the experimental study. To make fair comparisons, we fix all parameters that are not related to the model architecture.

We consider the up-scaling factors commonly used in most SISR methods, including $\times 2$, $\times 3$, and $\times 4$. The questions presented in Section 4.1 will be investigated in details for each of these factors. Even though it has been proven that a single model is competent for multiple scales [88], we empirically found that such a model requires high complexity and its training takes long time to converge. In our studies, we train the CNN models for these factors separately.

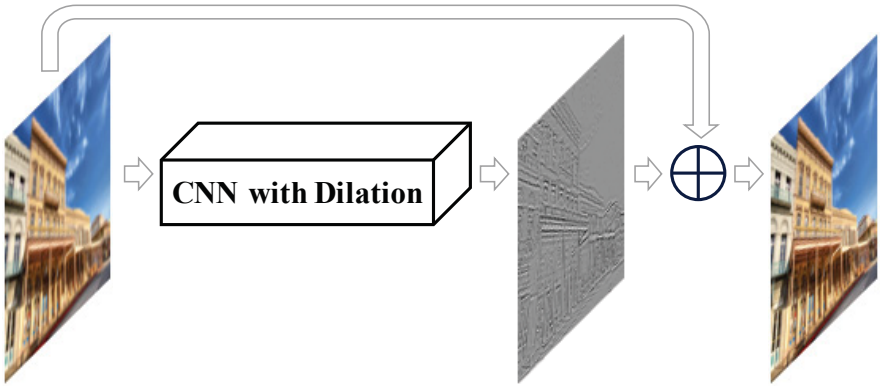


Figure 4.2: Basic CNN architecture. The CNN model with dilation will be specified in each individual task.

The training data for all studies is the 291 images as in [88, 143]. Data augmentation, such as flip and rotation, is used. During training, the samples of size 80×80 are randomly cropped from the images on-the-fly. The test data includes Set5 [10], Set14 [192], the Berkeley Segmentation 100 test images (BSD100) [116], and Urban100 [74]. We use Bicubic interpolation to generate the LR images.

Only the luminance channel of images are used to train and test the models, while the chromatic channels are up-scaled via Bicubic interpolation. As pointed out by Kim et al. [88], training becomes easier when the CNN model predicts the residual between the ground truth and the Bicubic interpolated images, rather than directly predict the HR image. We follow such a residual learning strategy to save training time. Thus, our CNN models take as input an 80×80-sized Bicubic interpolated image, and output a residual image with the same size. The HR result is then estimated by a summation operation. The general architecture is depicted in Fig. 4.2, where the CNN model is constructed according to the specific settings in Sections 4.4 and 4.5.

Following most SISR methods [36, 88], we employ the Euclidean distance (equivalent to MSE) as the training objective:

$$\begin{aligned} \ell(x_{re}, \hat{x}) &= \|x_{re} - \hat{x}\|^2, \\ x_{re} &= x - x_{bi}, \end{aligned} \tag{4.5}$$

where x_{re}, $\hat{x}$, x, and x_{bi} are the ground truth residual image, the estimated

Table 4.1: Training parameters.

Parameter	Value
batch size	64
optimization algorithm	SGD
initial learning rate	0.1
learning rate update policy	step
decreasing factor of learning rate	0.1
step size for decreasing learning rate	70000
momentum	0.9
weight decay	0.0001
clip gradients	0.1
max number of iterations	100000
learning rate multiplier for filter parameters	1
learning rate multiplier for bias terms	0.1

residual image, the ground truth HR image, and the bicubic interpolated HR image, respectively. We use the Caffe package [81] to implement the training and test phases. The necessary training parameters are summarised in Table 4.1. The gradient clipping technique is employed, which was shown to be beneficial for training CNN networks in SISR task [88]. To avoid numerical instability in float-type computation, we normalise the pixel range of the images from $[0, 255]$ to $[0, 1]$. The maximal iterations for training and the step size for decreasing learning rate are set according to experiences. To verify that training can converge under such settings, we plot the curves of test performances on Set5 which are produced by 11-depth models in different SISR tasks, as shown in Fig. 4.3.

We note that in SISR, the HR image pixels have close values to the LR image pixels, which means that the output values of the network are close to the input values. Thus, the bias terms in convolutional layers are not expected to have large values. Regarding this, we set the learning rate of all bias terms as 0.1 of the learning rate of filter parameters, as indicated by the last row of Table 4.1.

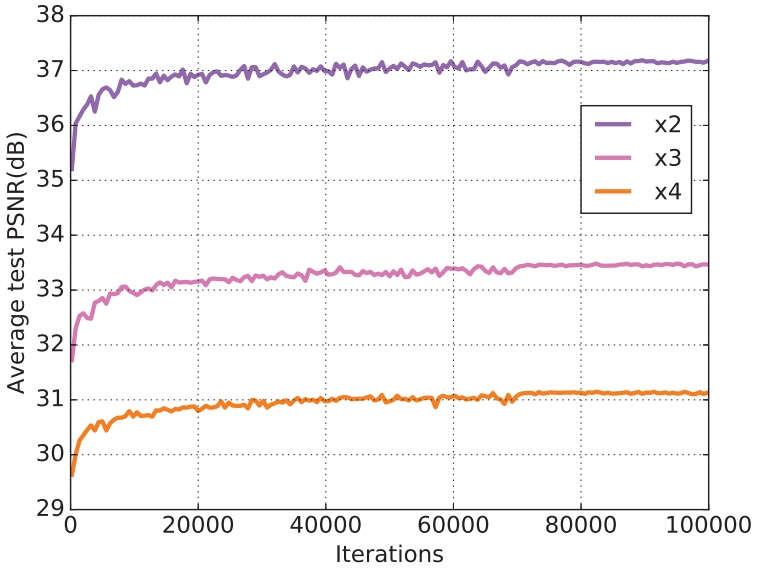


Figure 4.3: The curves of test performances on Set5 for $\times 2$, $\times 3$, and $\times 4$ tasks.

4.4 Effects of Receptive Field Size on SISR

In this section, we investigate the effects of different receptive field sizes on the SISR performance. For comparison, the model nonlinearity is fixed, which is achieved by keeping the model depth unchanged. In this condition, we employ the dilated convolutional layers which can exponentially increase the receptive field of a CNN model, as introduced in Section 4.2. In the following, we will use DC to denote the dilated convolution for concise presentation.

We begin by describing the notations that facilitate us to specify the CNN architectures designed in the following. We use LmDn to denote that a CNN model contains $m+1$ convolutional layers among which n DC layers are inserted. The one additional convolutional layer (*i.e.* "+1") is used for reconstructing the single-channel output, which is topped on all CNN models.

In all convolutional layers including the dilated ones but except the last layer, we use 64 filters with the basic size of 3×3 (dilation enlarges the filter size according to the dilation factor). While a larger filter can increase the receptive field, high computational cost would be induced. 3×3 has been shown in different CNN-based applications [64, 88, 150] to be effective to encode local structures of

Table 4.2: Receptive field sizes of different dilated filters (3×3-basic size).

a	0	2	4	8	16
Receptive field size	3	5	9	17	33

the input while preserving small number of parameters.

To insert DC layers, we have two considerations: how to change the dilation factor and where to insert. Regarding the first one, we follow the dilation strategy which is consistent with the multi-scale wavelet decomposition, *i.e.* $a = 2^i$ for the i-th DC layer where $i = 1, ..., n$. For example, if $n = 3$, a will be set to 2, 4, and 8 for the first, second, and third DC layers, respectively. Even though a can be set arbitrarily to increase the receptive field, such as in image segmentation [25], their motivation is to exploit global contexts as much as possible for semantically understanding the category of each pixel. However, we intuitively realise that it is more important for SISR to analyse local structures rather than global contexts. Too large receptive field is not helpful for improving the SISR performance, as demonstrated in the following experiments. In our work, the maximal number of DC layers that we will insert is $n = 4$. The receptive field sizes of different dilated filters are listed in Table 4.2.

For the second concern, our experiences indicate that when the DC layers are inserted into the very beginning position (close to the input) or the very end position (close to the output) of the network, the performance is limited. It is possibly because 1) the nonlinear function expressed by the first few layers is not effective for analysing the local structures in a large region of the LR image, and 2) the reconstruction of a HR pixel should be performed on local structures rather than distant points on feature maps. In fact, when applying a CNN model to SISR, we can divide the model by half, and view the first half part as encoding phase and the second half part as decoding phase. Such an interpretation has been exploited in the prior work on stacked auto-encoders [169, 171] and Dong et al.'s work [36]. In this sense, we propose to insert the DC layers around the middle position of the CNN model. In specific, when LmDn is used, the DC layers are successively placed after the $\lfloor \frac{m+1-n}{2} \rfloor$-th layer.

Regarding the image quality assessment metric, we denote the assessment metric by $\mathcal{P}^s_{\text{model}}$, where the subscript indicates the used model and the superscript

Table 4.3: Architecture settings of the L10 CNN models.

Model	DC positions	Receptive Field Size
L10D0	-	23
L10D2	6, 7	31
L10D3	5, 6, 7	45
L10D4	4, 5, 6, 7	75

denotes the up-scaling factor. For example, $\mathcal{P}^2_{\text{bi}}$ means the quality produced by Bicubic interpolation in the $\times 2$ SISR task. The *improved quality* by method A with respect to (*w.r.t.*) B is defined as

$$\Delta_B \mathcal{P}^s_A = \mathcal{P}^s_A - \mathcal{P}^s_B. \tag{4.6}$$

In our work, we consider two metrics including peak-to-signal noise ratio (PSNR) and structural similarity (SSIM) [175]. In the following analysis, we use $\mathcal{P}^s_{\text{model}}$ to denote both cases of the metrics. A detailed explanation will be provided to avoid ambiguous understandings when necessary.

Given the above settings and definitions, we first evaluate the performances of the L10 models with different receptive field sizes.

4.4.1 L10 CNN models

For the L10 models, we choose four schemes to insert the DC layers, including L10D0, L10D2, L10D3, and L10D4. The inserted positions and the respective field sizes are summarised in Table 4.3, which indicates that various receptive field sizes are considered.

We first illustrate the average qualities on each of the test sets, as shown in Fig. 4.4. Since the models take as input the Bicubic interpolated images, we calculate the improved quality of each model *w.r.t.* Bicubic interpolation, *i.e.* $\Delta_{bi}\mathcal{P}^s_{\text{L*D*}}$, to show how much the models can boost the SISR performance, as marked in Fig. 4.4.

Fig. 4.4a shows that the model L10D2 achieves the best PSNR value. But only a marginal improvement can be observed compared with the others (in most cases less than 0.1dB). The second best model is L10D0 which performs almost

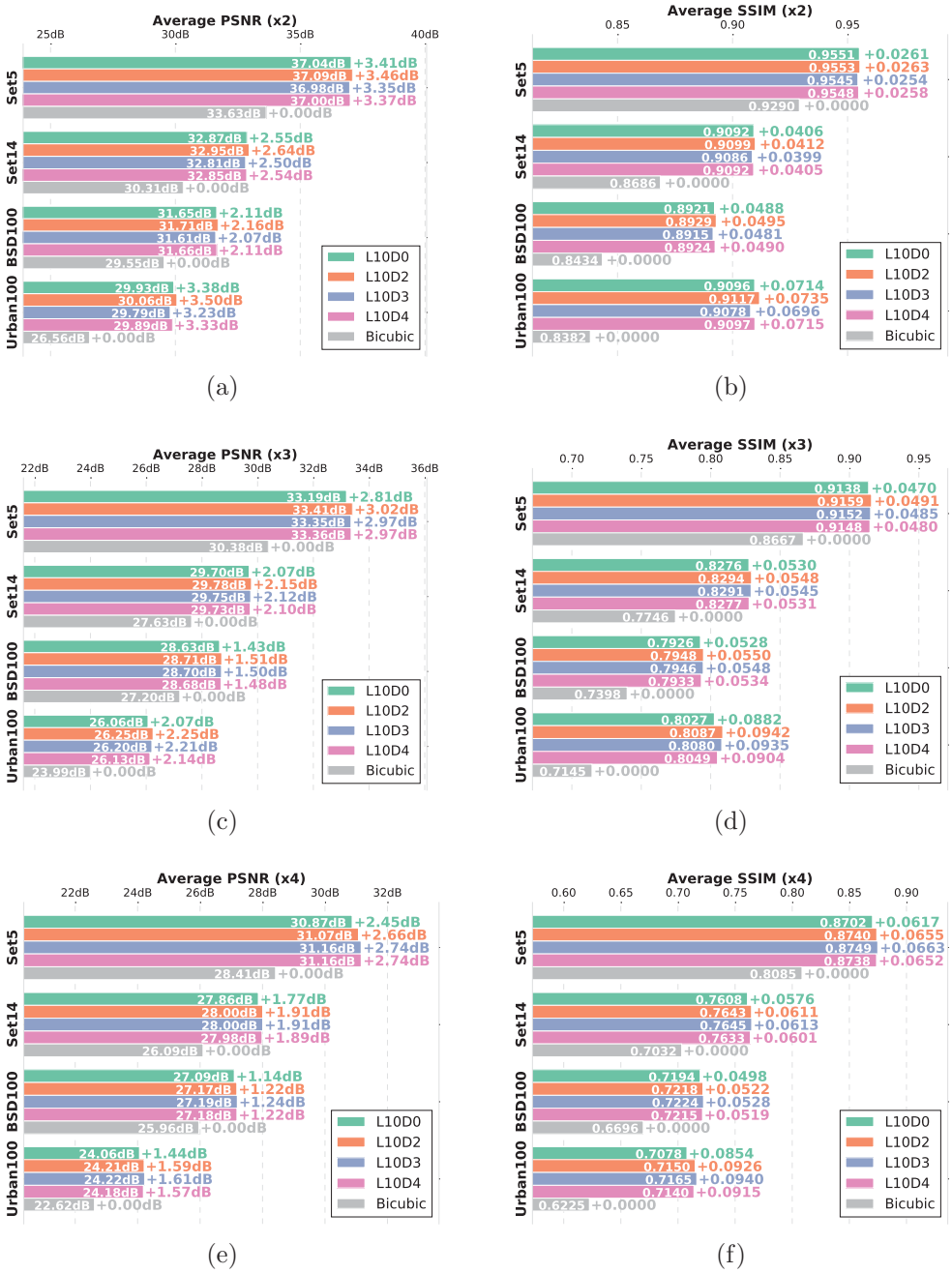


Figure 4.4: The SISR performances of the L10 models with different number of dilated convolutional layers. The up-scaling factor used in each comparison is marked in the top of the subfigures. In each dataset, the positive value on the right of each bar indicates the improved quality by the corresponding model *w.r.t.* Bicubic interpolation. The minimum at x-axis is adjusted for better visualisation of the differences between the performances.

the same as L10D2. An interesting case is that L10D3 is outperformed by both L10D2 and L10D4, which is possibly because the receptive field size of 45 is not a proper choice under the setting of L10. Similar phenomenon can also be observed from the SSIM metric, as shown in Fig. 4.4b. These results suggest that when using a L10 model, 1) increasing the receptive field size does not necessarily benefit for the $\times 2$ SISR task, and 2) a receptive field size between 23 to 31 is large enough to produce a pleasing performance here. This tells us that for the up-scaling factor of 2, a HR pixel is highly correlated with its surrounding pixels, and it is better to perform the HR estimation locally. Getting more pixels involved in estimation may degrade the final performance, given that the model nonlinearity is specified by L10.

In Fig. 4.4c, it is observed that L10D2 surpasses L10D0 by 0.1$\sim$0.2dB in all datasets, and is the best one in setting of $\times 3$. Focusing on the dilated models, we see that the performance monotonously decreases when the receptive field size increases. In spite of this, all the dilated models (*i.e.* L10D2, L10D3, L10D4) perform better than L10D0. The results on SSIM metric in Fig. 4.4d depict similar phenomenons. We observe from those figures that in the $\times 3$ SISR task, a large receptive field size is preferred, *e.g.* 31 (although it may be not the optimum); and in addition, a larger receptive field size may bring deficiencies in performance, which can be compensated by increasing the model depth.

In the SISR task of $\times 4$, Figs. 4.4e and 4.4f illustrate that the receptive field size of 45 (specified by L10D3) is the best choice among the others. When smaller than 45, the size is positively correlated with the performance; see L10D0, L10D2 and L10D3. But the fourth DC layer does not help to improve the performance further, as in the case of L10D4. These results advise that the $\times 4$ SISR task requires a large receptive field size to cover more image contexts for the reconstruction of a single HR pixel. This is reasonable because much valuable information is lost in the down-sampled images for $\times 4$, especially in the texture regions. The restoration thus needs more detailed information in distant regions, not only in a local region surrounding the HR pixel.

By comparing the results of $\times 4$ and $\times 3$, we see that $\Delta_{\text{L10D0}}\mathcal{P}^4_{\text{L10D3}} > \Delta_{\text{L10D0}}\mathcal{P}^3_{\text{L10D2}}$ for both PSNR and SSIM in each dataset, where L10D3 and L10D2 are the best models in the respective tasks. Similar effects can also be observed between $\times 3$

Table 4.4: Comparison of improved quality $\Delta_{\text{L10D0}}\mathcal{P}^{s}_{\text{best}}$ for the L10 models.

	$\Delta_{\text{L10D0}}\mathcal{P}^{2}_{\text{L10D2}}$	$\Delta_{\text{L10D0}}\mathcal{P}^{3}_{\text{L10D2}}$	$\Delta_{\text{L10D0}}\mathcal{P}^{4}_{\text{L10D4}}$
Set5	0.05/0.0002	0.21/0.0021	**0.29/0.0046**
Set14	0.09/0.0006	0.08/0.0018	**0.14/0.0037**
BSD100	0.05/0.0007	0.08/0.0022	**0.10/0.0030**
Urban100	0.12/0.0021	**0.18**/0.0060	0.17/**0.0086**

Table 4.5: Architecture settings of the L6 and L20 CNN models.

Model	DC positions	Receptive Field Size
L6D0	-	15
L6D2	3, 4	23
L6D3	3, 4, 5	37
L6D4	2, 3, 4, 5	67
L20D0	-	43
L20D2	10, 11	51
L20D3	10, 11, 12	65
L20D4	9, 10, 11, 12	95

and $\times 2$ according to $\Delta_{\text{L10D0}}\mathcal{P}^{3}_{\text{L10D2}} > \Delta_{\text{L10D0}}\mathcal{P}^{2}_{\text{L10D2}}$. The quantitative comparison can be found in Table 4.4, from which we know that enlarging the receptive field size can help more for the $\times 4$ SISR task than for both the $\times 2$ and $\times 3$ SISR tasks.

After analysing Fig. 4.4 and Table 4.4, we obtain the following remarks. On one hand, given a L10 model, a HR pixel can be effectively estimated based on a region of size 23$\sim$31 for $\times 2$ task, of size 31 for $\times 3$ task, and of size 45 for $\times 4$ task. Thus, it is beneficial to increase the receptive field size on the basis of a L10D0 model. On the other hand, when the degradation of an image gets more severe, a large receptive field (within a certain range) is more helpful than a small one for producing the top SISR performance.

4.4.2 L6 and L20 CNN models

In this subsection, we conduct more experiments to verify the remarks obtained by using L10 models in last subsection. In specific, shallower models L6Dn and deeper models L20Dn are selected. The detailed architecture settings are summarised in Table 4.5. Note that L20D4 has an overlarge receptive field, and we

accordingly change the size of training samples from 80×80 to 95×95 when training this model. The resultant performances of each model on the four test sets are illustrated in Figs. 4.5 and 4.6.

In the case of L6 models (Fig. 4.5), we observe similar effects to the case of L10 models, which state that L6D2 is the best model for both $\times 2$ and $\times 3$ tasks, while L6D3 is the best for $\times 4$. The overall performances of L6 models are slightly lower than that of L10 models due to insufficient nonlinearity. An obvious phenomenon is that the performance of L6D4 is worse, by a noticeable margin, than that of L6D0 in all datasets and all SR tasks, which is different from the case of L10. We foretell that increasing the receptive field size can involve more contextual information for SR, but a model of high nonlinearity is required. Apparently, the nonlinearity of L6 models is not strong enough to handle such a case.

Regarding the L20 models, Fig. 4.6 shows that the performance differences between L20D0, L20D2, and L20D3 are marginal. We notice that the results of PSNR and SSIM are not always consistent. For example, in $\times 2$ task, L20D2 achieves the best average PSNR while L20D0 has the best average SSIM; in $\times 3$ task, L20D3 and L20D2 are the best for PSNR and SSIM, respectively. This inconsistency may be caused by the training loss, *i.e.* MSE, which prefers high PSNR instead of high SSIM. L20D4 with an overlarge receptive field fails to produce a higher performance compared with L20D0.

We compute the values of the improved quality $\Delta_{\mathrm{L}m\mathrm{D0}}\mathcal{P}_{\mathrm{best}}$ for both L6 and L20, which are listed in Tables 4.6 and 4.7, respectively. The best models for L20 are selected according to the PSNR metric (since the improvement is not significant, we do not mark the best values). By comparing the improved qualities for L6, L10, and L20, we find that the receptive field size is an important factor for SISR especially when the model depth is not sufficient. While the model is very deep, the receptive field size characterised by the stack of 3×3 filters may be saturated, in which case the performance cannot be substantially improved by increasing the size further.

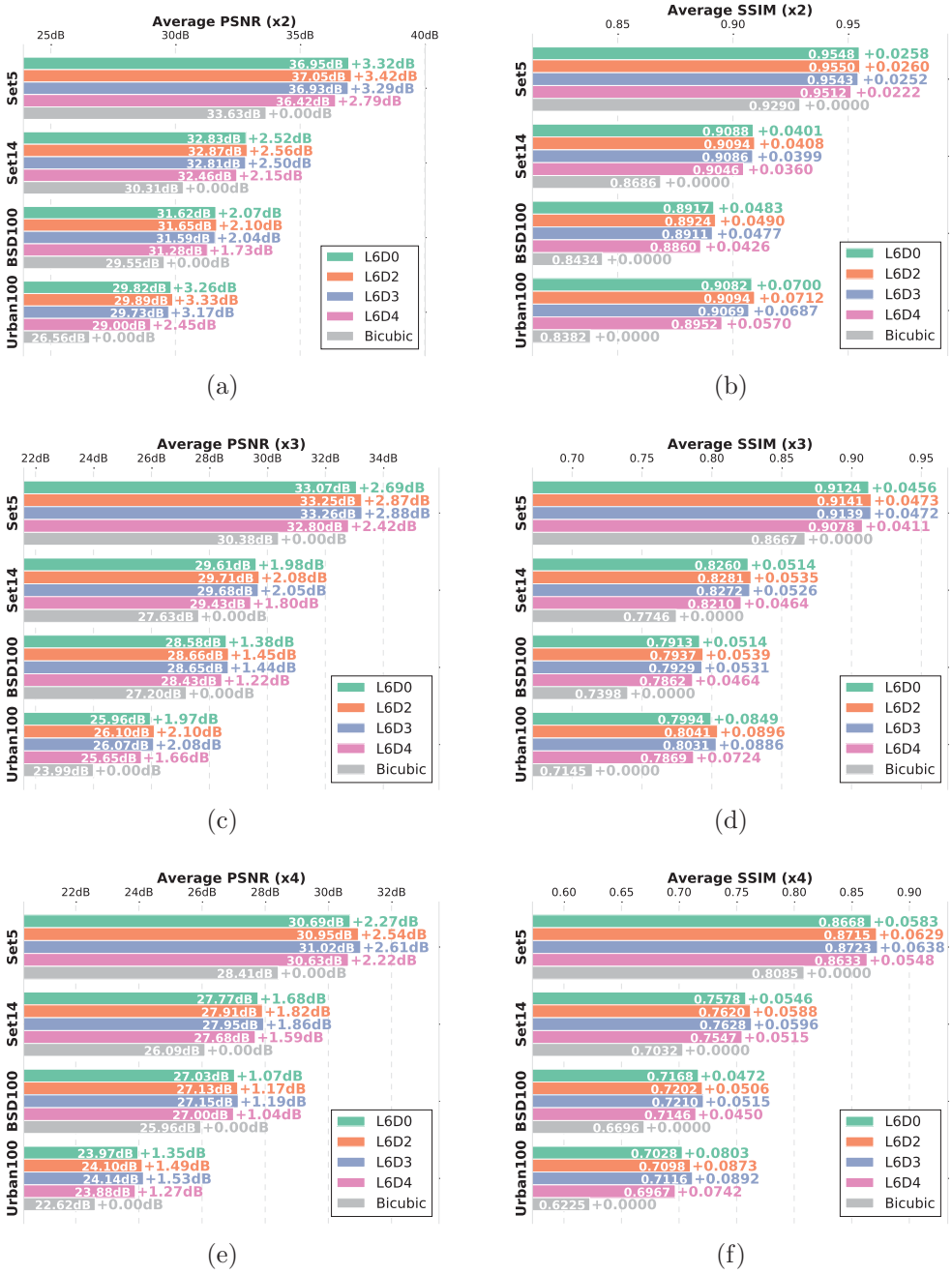


Figure 4.5: The SISR performances of the L6 models with different number of dilated convolutional layers.

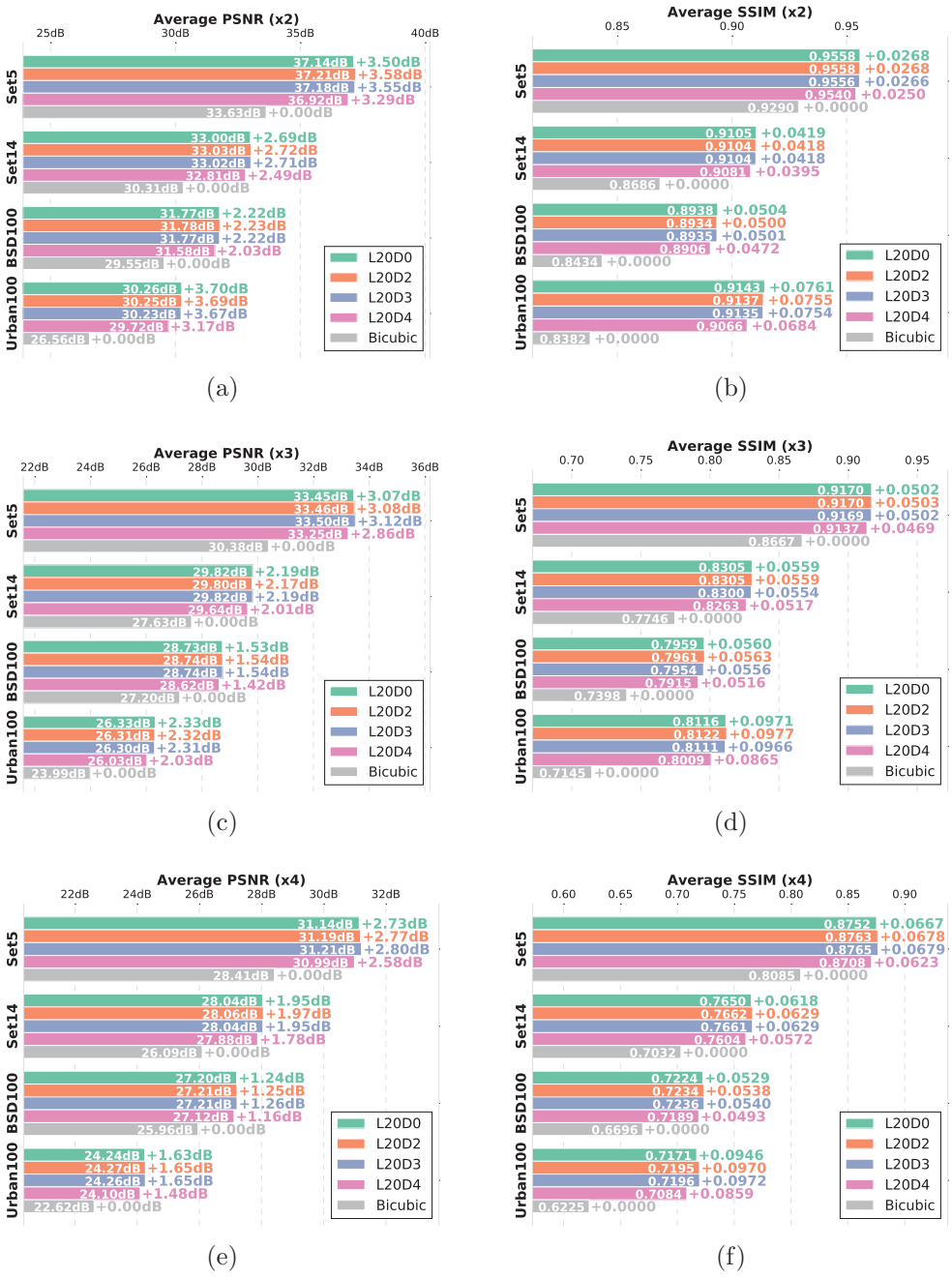


Figure 4.6: The SISR performances of the L20 models with different number of dilated convolutional layers.

Table 4.6: Comparison of improved quality $\Delta_{\text{L6D0}}\mathcal{P}^{s}_{\text{best}}$ for the L6 models.

	$\Delta_{\text{L6D0}}\mathcal{P}^{2}_{\text{L6D2}}$	$\Delta_{\text{L6D0}}\mathcal{P}^{3}_{\text{L6D2}}$	$\Delta_{\text{L6D0}}\mathcal{P}^{4}_{\text{L6D4}}$
Set5	0.10/0.0002	0.18/0.0017	**0.34/0.0055**
Set14	0.04/0.0007	0.10/0.0021	**0.18/0.0050**
BSD100	0.03/0.0007	0.07/0.0025	**0.12/0.0043**
Urban100	0.07/0.0012	0.13/0.0047	**0.18/0.0089**

Table 4.7: Comparison of improved quality $\Delta_{\text{L20D0}}\mathcal{P}^{s}_{\text{best}}$ for the L20 models.

	$\Delta_{\text{L20D0}}\mathcal{P}^{2}_{\text{L20D2}}$	$\Delta_{\text{L20D0}}\mathcal{P}^{3}_{\text{L20D3}}$	$\Delta_{\text{L20D0}}\mathcal{P}^{4}_{\text{L20D4}}$
Set5	0.08/0.0000	0.05/0.0000	0.07/0.0012
Set14	0.03/-0.0001	0.00/-0.0005	0.00/0.0011
BSD100	0.01/-0.0004	0.01/-0.0004	0.02/0.0011
Urban100	-0.01/-0.0006	-0.02/-0.0005	0.02/0.0026

4.4.3 An instance-level investigation

While the above-mentioned analysis concern about the effects of receptive field size on different up-scaling factors, we next investigate the effects on each instance of BSD100 and Urban100. This investigation is to explore what kind of images benefits more from increasing the receptive field size. Specifically, we employ the *local entropy* to characterise the image type. As introduced by Kadir and Brady [85], given a local region R of a grey-level image and the pixel descriptor $D \in \{d_1, ..., d_n\}$, the local entropy is calculated as

$$H_{D,R} = -\sum_{i} P_{D,R}(d_i) \log_2 P_{D,R}(d_i), \tag{4.7}$$

where $P_{D,R}(d_i)$ is the probability of the descriptor D taking the value d_i in the local region R. This quantity has nice properties to distinguish different kinds of regions in an image; for example, texture, edge, and flat regions have high, moderate, and low values of the local entropy, respectively, which can be observed from the examples in Fig. 4.7. We regard the average local entropy over the whole image, denoted by $\bar{H}_D$, as an indicator of whether the image has more textures, more edges, or more flat areas. Then, the goal is to find the correlation between the average local entropy and the improved quality by increasing the receptive

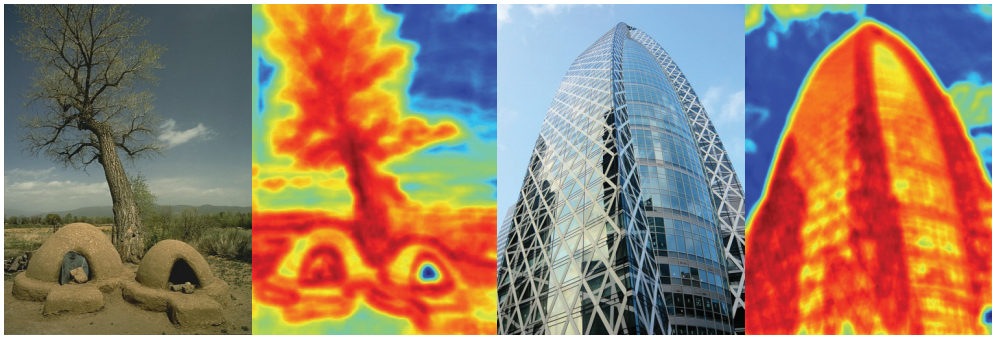


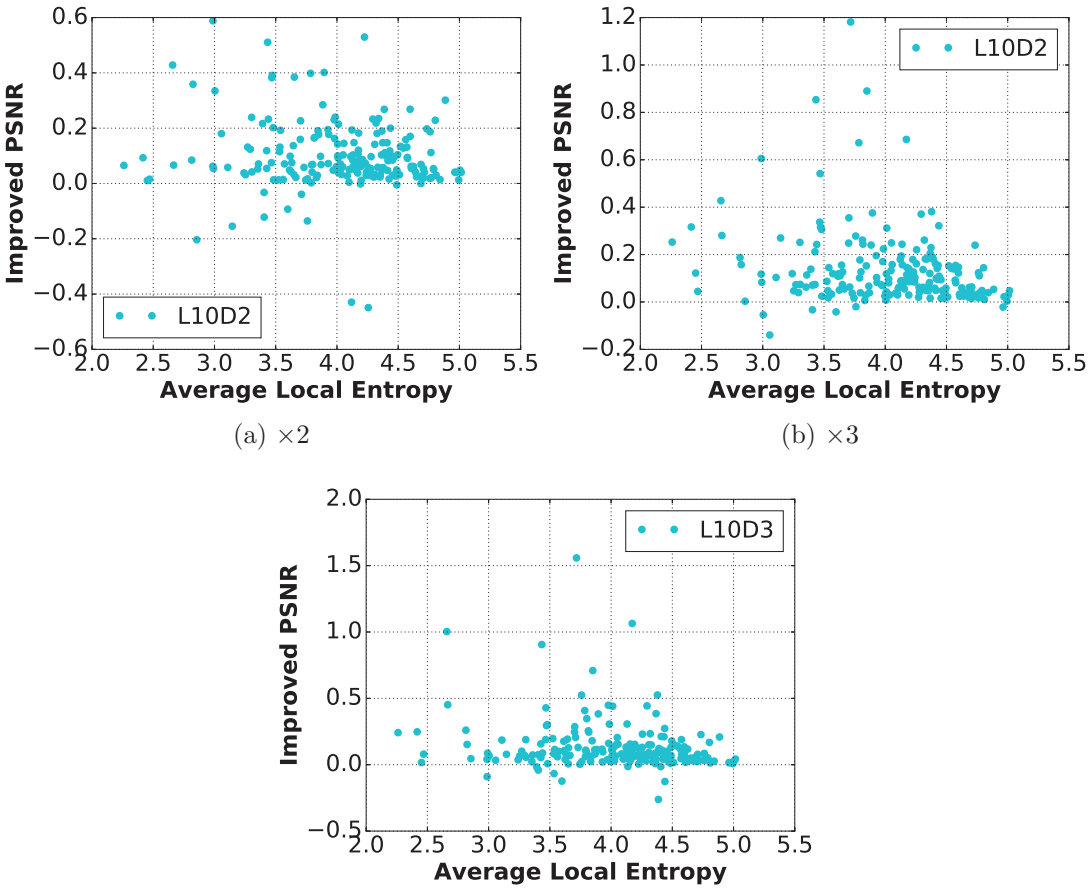
Figure 4.7: Examples of local entropy.

field size. In each SISR task, the value of the pair $(\Delta_{\text{L10D0}}\mathcal{P}^s_{\text{best}}, \bar{H}_D)$ is calculated for each image in BSD100 and Urban100 (specifically, PSNR is calculated). The results in Fig. 4.8 illustrate that for all tasks, $\Delta_{\text{L10D0}}\mathcal{P}^s_{\text{best}}$ has scattered large values when $\bar{H}_D$ is low. With the increase of $\bar{H}_D$, $\Delta_{\text{L10D0}}\mathcal{P}^s_{\text{best}}$ tends to shrink to small values (even zero). This implies that increasing the receptive field size works well in the flat and edge regions that possess low local entropy values, but it is limited to tackle highly textured image regions.

We select a set of images that achieve the highest and the lowest values of $\Delta_{\text{L10D0}}\mathcal{P}^s_{\text{best}}$, as shown in Fig. 4.9. The bad examples contain many textures in the residual images that are not recovered, and it is worth noting that the residual images are highly correlated with the local entropy images. One reason for the failure is that it is difficult to model the complex structures in texture areas by using limited information in LR images. Another reason is the training loss, *i.e.* MSE. As indicated in [99], the solution induced by MSE loss is a pixel-wise average of possible HR patches on a natural image manifold, leading to blurry effects in the restored images.

Remarks on Receptive Field Size:

- Within a certain range, increasing the receptive field size can always help improve the SISR performance.
- Receptive field size is an important factor which can compensate the deficiency caused by insufficient model nonlinearity.
- A larger receptive field size is required when the image degradation gets



(a) ×2 (b) ×3

(c) ×4

Figure 4.8: Improved PSNR $\Delta_{\text{L10D0}}\mathcal{P}^s_{\text{best}}$ *v.s.* average local entropy for each image in BSD100 and Urban100. In each case, the best model (L10D2 for ×2, L10D2 for ×3, and L10D3 for ×4) is used for evaluation.

more severe.

4.5 Effects of Model Depth on SISR

This section focuses on a detailed analysis of how model depth affects the SISR performance. To make a fair comparison, the models used in this section have the same receptive field size. For this purpose, we propose to employ the convolutional layer with 1×1 filter size, which is an effective way to arbitrarily increase the

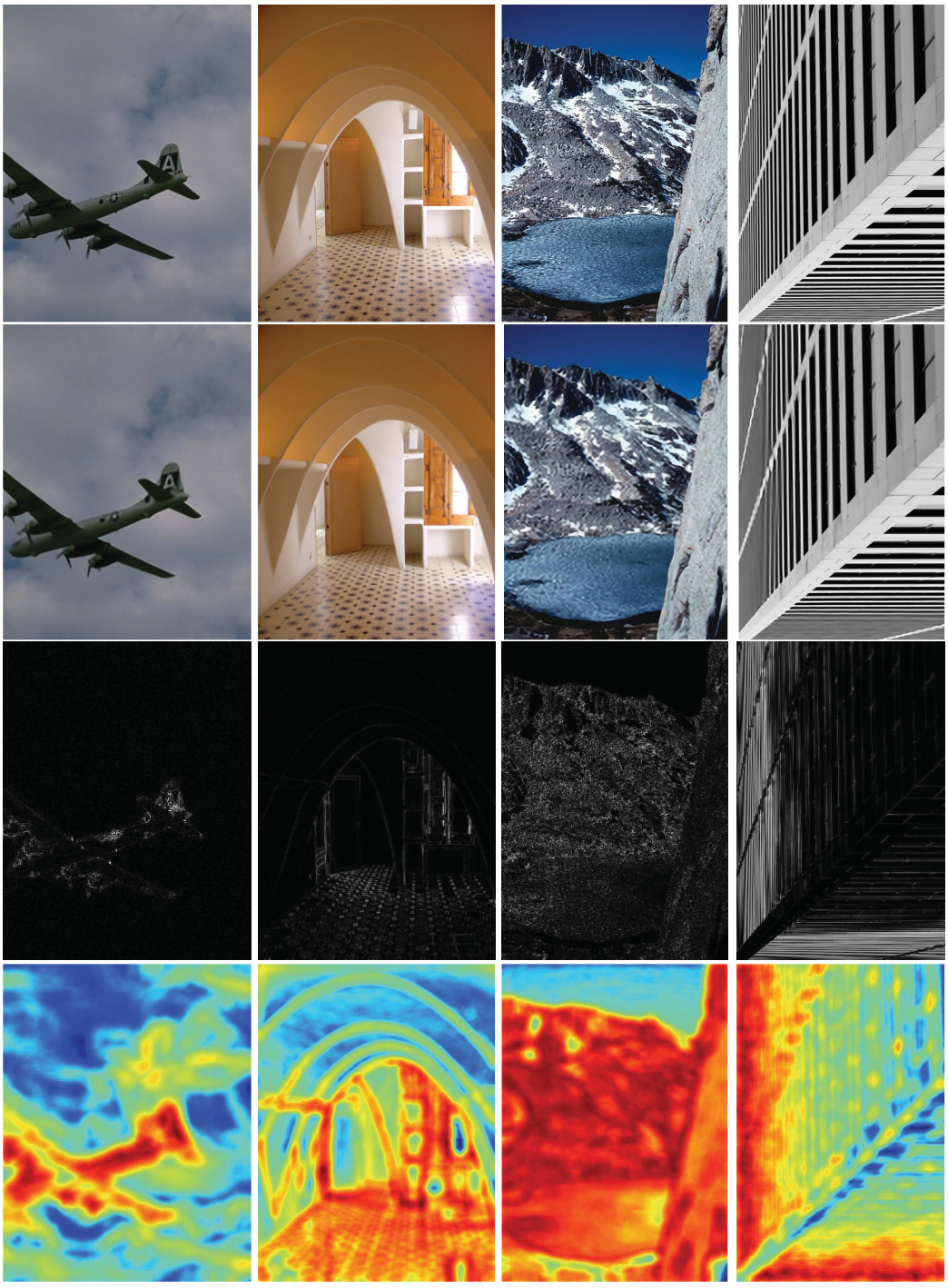


Figure 4.9: SR examples of L10D2 for $\times 3$. From top to bottom: original HR image, restored image, residual image, and local entropy image. The left two columns are the examples that achieve the highest $\Delta_{\mathrm{L10D0}}\mathcal{P}^3_{\mathrm{L10D2}}$ in BSD100 and Urban100, respectively. The right two columns achieve the lowest $\Delta_{\mathrm{L10D0}}\mathcal{P}^3_{\mathrm{L10D2}}$ in the two datasets.

model nonlinearity without changing the receptive field size. In the following, the convolutional layer with $k \times k$ filter size is denoted by $k \times k$ convolutional layer for concise presentation.

To begin with, we modify the model notations to incorporate with the newly introduced 1×1 convolutional layers. Specifically, the notation LmDnPo is used to denote a CNN model with $m + 1$ 3×3 convolutional layers and o 1×1 convolutional layers, among which there are n DC layers. The total number of convolutional layers in this model is $m + o + 1$. Similar to the settings in previous section, all convolutional layers except the reconstruction layer employ 64 filters. The setting of the dilation factor follows Table 4.2.

One concern about designing a LmDnPo model is where to insert the 1×1 convolutional layers. Existing studies [37, 149, 189] inform us that a SISR process is a combination of LR feature encoding and HR feature decoding. Intuitively, in the architecture depicted by Fig. 4.2, we regard that the convolutional layers close to the input focus on LR feature encoding, while the convolutional layers close to the output are responsible for HR feature decoding. Since HR feature space is more complex than LR feature space, more nonlinear activation functions (*i.e.* ReLU) are required for HR feature analysis. In this sense, the CNN models are designed by inserting the 1×1 convolutional layers after the L 3×3 convolutional layers. Such a design allows that the contextual information in the receptive field can be completely extracted before the 1×1 convolutions.

To analyse the effects of model depth, we are particularly interested in two cases, the models with and without a sufficient receptive field size, which are discussed separately.

4.5.1 Insufficient receptive field size

The experiments in this subsection is to examine whether it is useful to apply more nonlinear computations on limited contextual information. We select different models in different SISR tasks to ensure that the receptive field size in each task is insufficient. Specifically, according to the results in Section 4.4, we select L6D0 (or L6D0P0) as the basic model in $\times 2$ task, which is compared with L6D0P4 and L6D0P14. L10D0 (or L10D0P0) is used as the basic model in both $\times 3$ and $\times 4$

Table 4.8: Architecture settings of the LmD0Po CNN models.

Model	Receptive Field Size	Depth
L6D0P0	15	7
L6D0P4	15	11
L6D0P14	15	21
L10D0P0	23	11
L10D0P5	23	16
L10D0P10	23	21

tasks, which is compared with L10D0P5 and L10D0P10. The receptive field sizes and model depths are summarised in Table 4.8.

The comparison of the performances produced by different models is exhibited in Fig. 4.10. In the case of $\times 2$, it is noticed that L6D0P4 performs slightly better than L6D0P0 and L6D0P14. Similarly in the $\times 3$ task, L10D0P5 achieves the best performance among the others. In the case of $\times 4$, L10D0P5 and L10D0P10 perform almost the same, and are both better than L10D0P0. Overall, the largest improvements *w.r.t.* LmD0P0 in $\times 3$ and $\times 4$ exceed the largest improvements in $\times 2$. But the overall improvements in all cases are marginal and not very attractive.

These results suggest that given a model with insufficient receptive field size, increasing the model depth only slightly improves the SISR performance. An over-deep model without viewing adequate contextual information can even produce degraded performance (see L6D0P14 in $\times 2$ and L10D0P10 in $\times 3$). It is possibly due to that a long stack of 1×1 convolutional layers hinders the training process to produce a good model. In addition, when the information is lost significantly as in $\times 4$, a deep architecture is preferred. This is reasonable because the LR space structure is simple, whereas the HR space structure is complex. The significant differences between LR and HR spaces result in a highly nonlinear point-to-point mapping function from LR to HR spaces.

4.5.2 Sufficient receptive field size

In this subsection, we investigate how the model depth affects the SISR performance given the models with sufficient views of contextual information. We first

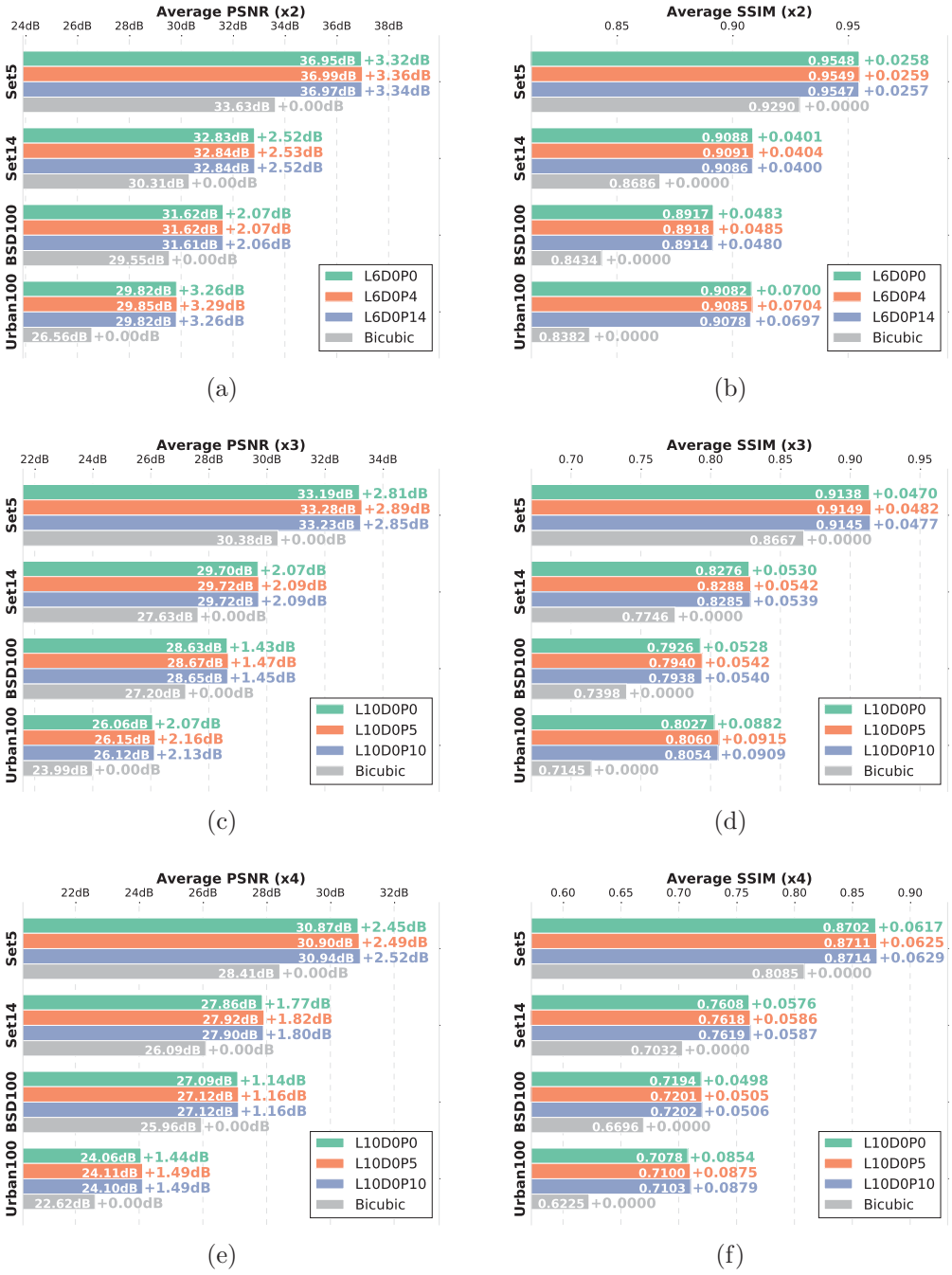


Figure 4.10: The SISR performances of the LmD0Po models for the investigation of model depth under the setting of an insufficient receptive field size.

Table 4.9: Architecture settings of the L6DnPo CNN models.

Model	DC positions	Receptive Field Size	Depth
L6D2P0	3, 4	23	7
L6D2P4	3, 4	23	11
L6D2P14	3, 4	23	21
L6D3P0	3, 4, 5	37	7
L6D3P4	3, 4, 5	37	11
L6D3P14	3, 4, 5	37	21

Table 4.10: Architecture settings of the L10DnPo CNN models.

Model	DC positions	Receptive Field Size	Depth
L10D0P0	-	23	11
L10D0P5	-	23	16
L10D0P10	-	23	21
L10D2P0	6, 7	31	11
L10D2P5	6, 7	31	16
L10D2P10	6, 7	31	21
L10D3P0	6, 7, 8	45	11
L10D3P5	6, 7, 8	45	16
L10D3P10	6, 7, 8	45	21

set up an experiment by using the basic shallow models, including L6D2P0 for $\times 2$ task and L6D3P0 for $\times 3$ and $\times 4$ tasks. Deeper models are constructed by stacking 4 and 14 1×1 convolutional layers. The model details can be found in Table 4.9.

The second experiment is to investigate whether the best L10 model in each SR task can be further improved by increasing model depth. The basic models used here are L10D0P0 for $\times 2$ task, L10D2P0 for $\times 3$ task, and L10D3P0 for $\times 4$ task. We do not use the L20 models because our experiences indicate that an over-deep model (*e.g.* with more than 30 layers) takes very long time for training. As previously, we choose two policies to increase the depth: adding 5 and 10 1×1 convolutional layers. The corresponding architecture settings are detailed in Table 4.10.

Figs. 4.11 and 4.12 illustrates the obtained performances, showing that only

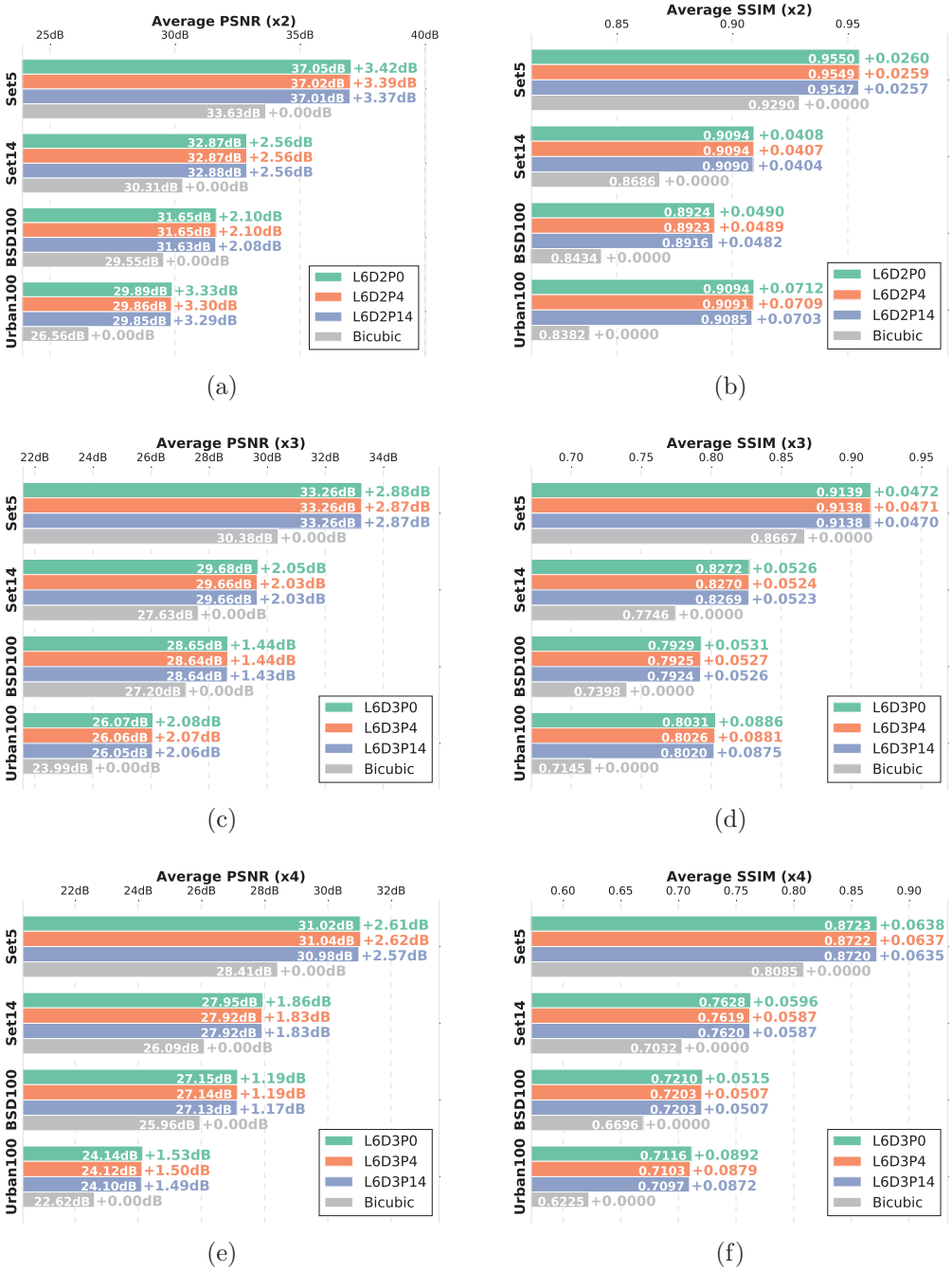


Figure 4.11: The SISR performances of the L6DnPo models for the investigation of model depth under the setting of a sufficient receptive field size.

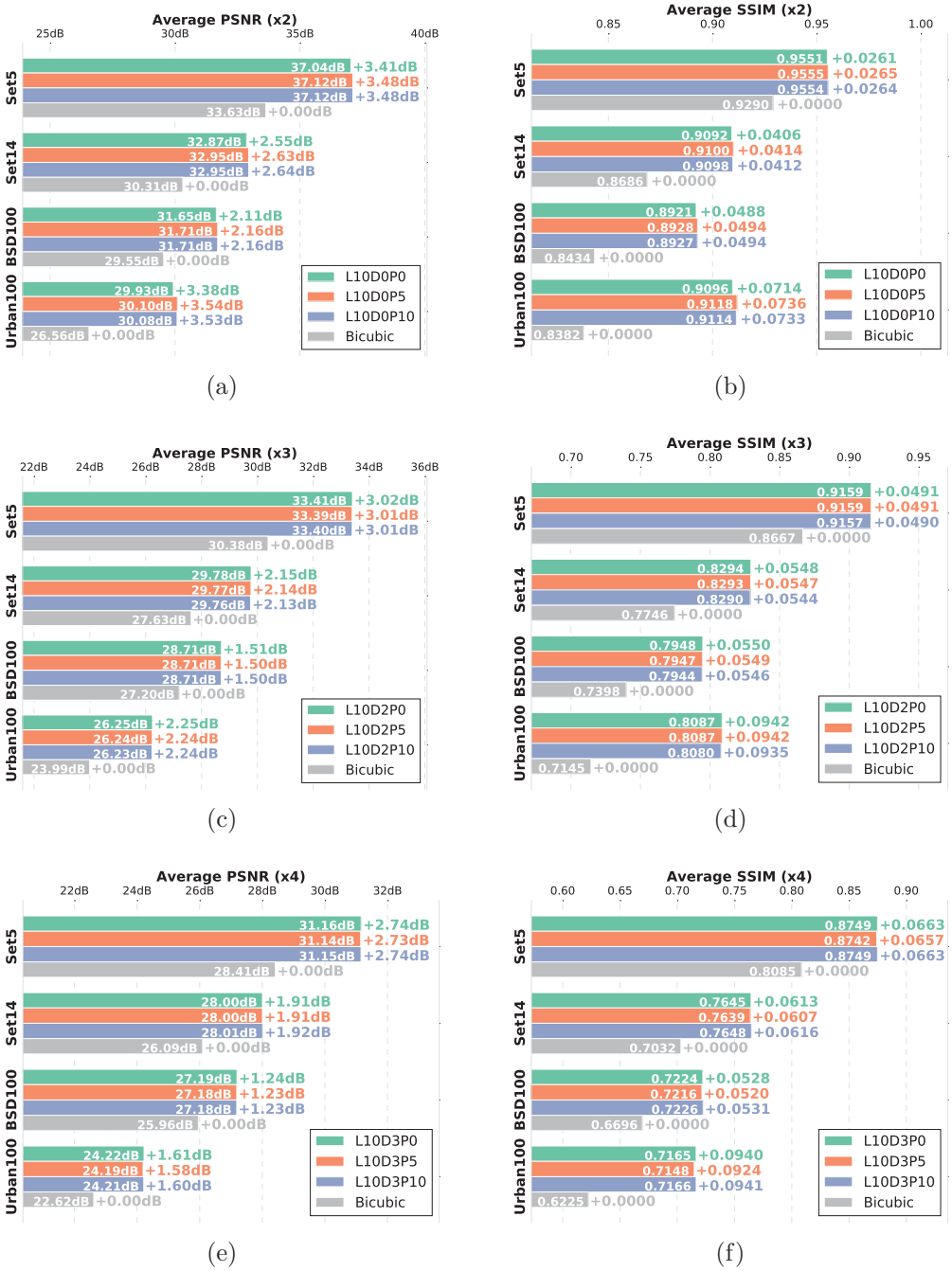


Figure 4.12: The SISR performances of the L10D*n*P*o* models for the investigation of model depth under the setting of a sufficient receptive field size.

the $\times 2$ task can benefit slightly from increasing the depth of L10D0P0. All other cases fail to show noticeable performance improvement when adding the 1×1 convolutional layers.

An explanation for the above observation is that the performances of the selected basic models may have reached the limit, and the induced nonlinearity is competent to analyse the contextual information viewed by the models. By comparing the values in Figs. 4.12 and 4.4, it is easily noticed that a L10Dn model can be improved more by enlarging receptive field than by increasing depth.

Remarks on Model Depth:

- It is not helpful for SISR to simply increase the model depth without changing the receptive field size.

4.6 Discussion

While the analysis in previous sections demonstrates that the receptive field size is more important than the model depth, we need to emphasise that the two factors are mutualistic within a certain range of values. That is, the larger the context region is, the stronger nonlinearity the model requires, and *vice versa*.

To jointly evaluate the influences of different receptive field sizes and model depths, we test the previously trained models on BSD100 and Urban100. The acquired average PNSR results are listed in Tables 4.11, 4.12, and 4.13. Since there would be an extremely large amount of work to exhaustively evaluate all combinations of the pairs (receptive field size, model depth), we only list the performances of the previously selected models and leave the rest as "-". As seen in the tasks, the differences between the four values along each row where applicable are no larger than 0.11dB, and mostly around 0.02dB. But when comparing the values along the columns, a large variation can be noticed. The best values in all tasks are produced by the L20 models with appropriate receptive field sizes, and the $\times 4$ task requires larger receptive field than the $\times 2$ and $\times 3$ tasks. We also note that the performance of a L10 model can approach to that of the best L20 model. For example, in $\times 2$ the best L10 is less than the corresponding best L20

Table 4.11: Average PSNRs (dB) (×2) using different receptive field sizes (RFS) and model depths. The top 30% values are marked by bold. The top three values are marked by red, green, and blue, respectively.

RFS \ Depth	7	11	16	21
15	30.72	30.73	-	30.71
23	30.77	30.79	**30.90**	**30.89**
31	-	**30.88**	-	-
37	30.66	-	-	-
43	-	-	-	**31.01**
45	-	30.70	-	-
51	-	-	-	**31.01**
65	-	-	-	**31.00**
67	30.14	-	-	-
75	-	30.77	-	-
95	-	-	-	30.65

Table 4.12: Average PSNRs (dB) (×3) using different receptive field sizes (RFS) and model depths.

RFS \ Depth	7	11	16	21
15	27.27	-	-	-
23	27.38	27.35	27.41	27.39
31	-	**27.48**	**27.47**	**27.47**
37	27.36	27.35	-	27.34
43	-	-	-	**27.53**
45	-	**27.45**	-	-
51	-	-	-	**27.53**
65	-	-	-	**27.52**
67	27.04	-	-	-
75	-	27.41	-	-
95	-	-	-	27.32

Table 4.13: Average PSNRs (dB) (×4) using different receptive field sizes (RFS) and model depths.

RFS \ Depth	7	11	16	21
15	25.50	-	-	-
23	25.61	25.58	25.61	25.61
31	-	25.69	-	-
37	25.65	25.63	-	25.62
43	-	-	-	**25.72**
45	-	**25.71**	25.69	**25.70**
51	-	-	-	**25.74**
65	-	-	-	**25.74**
67	25.44	-	-	-
75	-	25.68	-	-
95	-	-	-	25.61

by 0.13dB; in ×3 the best L10 is less than the best L20 by 0.05dB; and in ×4 the best L10 is less than the best L20 by 0.03dB.

According to these results as well as the previous analyses, we can **conclude** the relationship between receptive field size and model depth *w.r.t* SISR, which states:

- Model depth is an essential element for SISR to produce a top performance, which must cooperate with a proper receptive field size.
- Given a fixed receptive field size, increasing model depth is no helpful. But given a fixed model depth, the performance of the model is sensitive to the size of receptive field.
- When image degradation is severe, a large receptive field is required, and otherwise, a deep model is expected.
- The restoration of flat and edge regions can benefit more from increasing receptive field size, whereas the restoration of textures benefits less, given that the training objective is MSE.

From the perspective of image contexts and structures, we understand that in flat or edge areas, a HR pixel is highly correlated with the pixels in its surrounding

region or along a specific direction. If the degradation of the nearby region is not serious, the HR pixel can be easily estimated based on the surrounding LR pixels by using a model with appropriate complexity. But if the degradation is severe, especially when the nearby LR pixels become dissimilar to the HR pixel, a large region of context information is expected, and a highly nonlinear restoration function is potentially required. On the other hand, in texture areas, the local structures are always complex. In such a case, we can expect a model with large receptive field size and high nonlinearity to exploit regular patterns in the texture region, which may produce a good performance. But if the textures are irregular, the model will inevitably fail, especially when the degradation is serious.

The issue of texture restoration is also a bottleneck encountered by modern SISR techniques that are based on MSE training loss. Even though high PSNR values can be obtained (possibly due to the good estimation of flat and edge regions), the resultant textures generally exhibit blurry effects and are thus visually unpleasing. The very recent SISR researches begin to focus on developing perceptual losses by which the HR estimation exhibits clear textures. Even tough the restored textures are sometimes not similar to those in the original HR image, the resultant image becomes more perceptually acceptable. While the studies in this chapter are based on MSE, the empirical findings are consistent with our intuitive understanding, and are applicable for designing perceptual loss-based architectures.

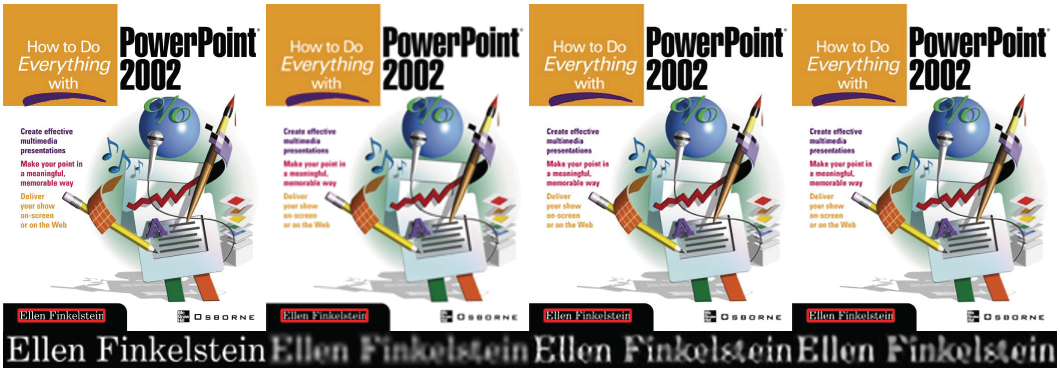
4.6.1 Comparison with state-of-the-art

We finally compare the best dilated L10 model with the non-dilated L10 and L20 models, as well as the methods including SRCNN [36], CSCN [174], FSRCNN [37] and VSDR [88]. The dilated model is named as dilated convolutional network for SR (DCNSR). The average performances are summarised in Table 4.14, and visual examples can be found in Figs. 4.13, 4.14, and 4.15.

As shown, VDSR, L20D0 and DCNSR_{10} are among the best competitors, and notably, their performances are very close to each other. This inspires us to use a 10-depth model with enlarged receptive field size to compete a 20-depth model in various applications, such that higher computational efficiency can be obtained

Table 4.14: Average PSNR(dB)/SSIM of different methods on Set5, Set14, BSD100, and Urban100.

Dataset	Scale	Bicubic	SRCNN	CSCN	FSRCNN	VDSR	L10D0	L20D0	$DCNSR_{10}$
Set5	×2	33.66/0.9299	36.66/0.9542	36.93/0.9552	37.00/0.9558	37.53/0.9587	37.04/0.9551	37.14/0.9558	37.09/0.9553
	×3	30.39/0.8682	32.75/0.9090	33.10/0.9144	33.16/0.9140	33.66/0.9213	33.19/0.9138	33.45/0.9170	33.41/0.9159
	×4	28.42/0.8104	30.48/0.8628	30.86/0.8732	30.71/0.8657	31.35/0.8838	30.87/0.8702	31.14/0.8752	31.07/0.8740
Set14	×2	30.24/0.8688	32.42/0.9063	32.56/0.9074	32.63/0.9088	33.03/0.9124	32.87/0.9092	33.00/0.9105	32.95/0.9099
	×3	27.55/0.7742	29.28/0.8209	29.41/0.8238	29.43/0.8242	29.77/0.8314	29.70/0.8276	29.82/0.8305	29.78/0.8294
	×4	26.00/0.7027	27.49/0.7503	27.64/0.7587	27.59/0.7535	28.01/0.7674	27.86/0.7608	28.04/0.7650	28.00/0.7643
BSD100	×2	29.56/0.8431	31.36/0.8879	31.40/0.8884	31.50/0.8909	31.90/0.8960	31.65/0.8921	31.77/0.8938	31.71/0.8929
	×3	27.21/0.7385	28.41/0.7863	28.50/0.7885	28.52/0.7900	28.82/0.7976	28.63/0.7926	28.73/0.7959	28.71/0.7948
	×4	25.96/0.6675	26.90/0.7101	27.03/0.7161	26.96/9,7139	27.29/0.7251	27.09/0.7194	27.20/0.7224	27.17/0.7218
Urban100	×2	26.88/0.8403	29.50/0.8946	29.27/0.9031	29.49/0.9035	30.76/0.9140	29.93/0.9096	30.26/0.9143	30.06/0.9117
	×3	24.46/0.7349	26.24/0.7989	25.77/0.7961	25.84/0.7945	27.14/0.8279	26.06/0.8027	26.33/0.8116	26.25/0.8087
	×4	23.14/0.6577	24.52/0.7221	23.96/0.7065	23.91/0.6976	25.18/0.7524	24.06/0.7078	24.24/0.7171	24.21/0.7150

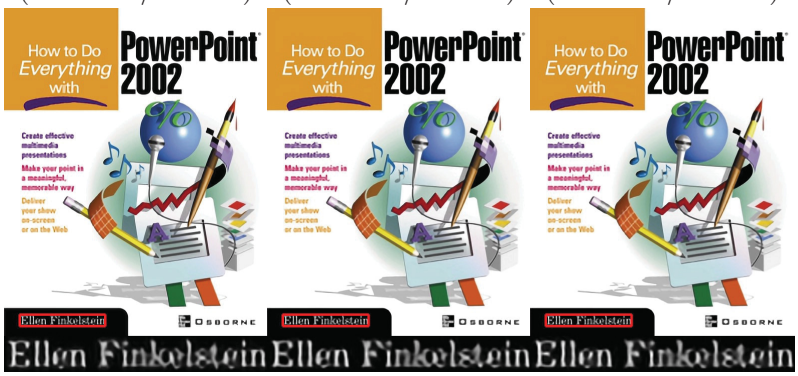


Original

Bicubic (23.76dB/0.8853)

SRCNN (27.11dB/0.9449)

CSCN (27.10dB/0.9545)



FSRCNN (27.23dB/0.9536)

L20D0 (27.95dB/0.9620)

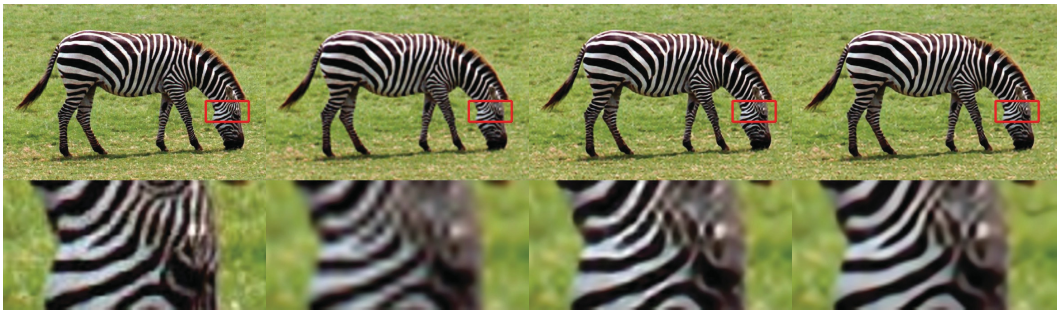
$DCNSR_{10}$ (27.94dB/0.9618)

Figure 4.13: SR examples with an up-scaling factor of 3. PSNR/SSIM values are marked under each subfigure.

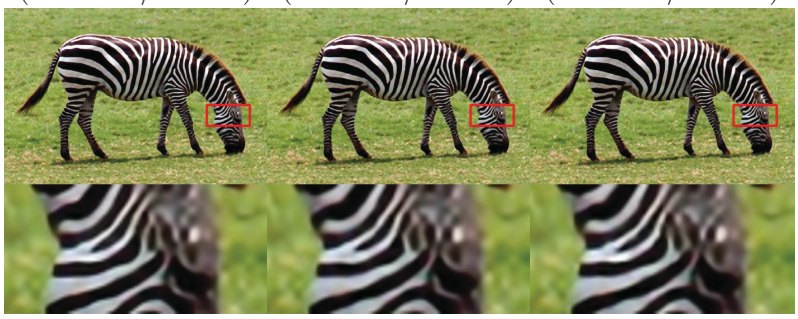
without sacrificing performance.

4.7 Conclusion

In this chapter, we present our investigation to answer the question: does SISR require a complex function or a wide region, or both, to produce good estimations of HR pixels? Accordingly we conduct a comprehensive investigation on two key factors of a deep architecture, namely receptive field size and model depth, which may affect the SISR performance. Specifically, it is proposed to employ the dilated convolutional layers which can enlarge the receptive field size without changing the model depth. Our findings state that given the model depth fixed,



Original | Bicubic (26.66dB/0.7944) | SRCNN (29.31dB/0.8531) | CSCN (29.45dB/0.8547)

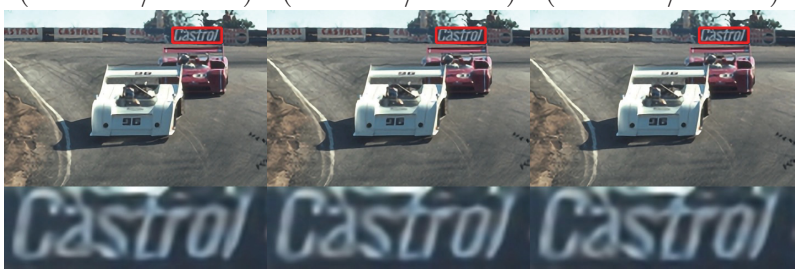


FSRCNN (29.50dB/0.8563) | L20D0 (29.65dB/0.8567) | $DCNSR_{10}$ (29.58dB/0.8559)

Figure 4.14: SR examples with an up-scaling factor of 3 (*cont.*). PSNR/SSIM values are marked under each subfigure.

the SISR performance is sensitive to the selection of the receptive field size, and that the more seriously the image is degraded, the larger receptive field size is required. In the studies on model depth, we propose to use 1×1 convolutional layers to strengthen the model nonlinearity. It is observed that there is no noticeable improvement on performance by simply stacking more convolutional layers without extending the receptive field. But in an investigation of the joint effects between receptive field size and model depth, we find that a deep architecture is necessary to produce the top performance, while the optimal receptive field size should be congruent with the model depth. It is also noticed that the model depth has more influence on performance when the image degradation is mild, and the receptive field size is more important when the degradation is severe. The proposed strategy of dilated convolution can also help reduce model depth while maintaining desirable performances, thus saving the computation cost. Further

Original | Bicubic (26.21dB/0.7348) | SRCNN (27.73dB/0.7837) | CSCN (27.18dB/0.7849)



FSRCNN (27.55dB/0.7908) | L20D0 (28.27dB/0.7955) | $DCNSR_{10}$ (28.24dB/0.7947)

Figure 4.15: SR examples with an up-scaling factor of 3 (*cont.*). PSNR/SSIM values are marked under each subfigure.

work will focus on exploiting more combining ways of dilated convolution layers to design more efficient SISR networks.

Chapter 5

Residual Learning in Non-Blind Image Deconvolution

It is always a challenging issue to develop a single model to handle multiple types of degradations, because the images under different degradations generally exhibit different characteristics of image textures, making the optimal model very complex. For example, in the non-blind image deconvolution task, a blurry image is correlated with the corresponding sharp image, the blur kernel, and the noise. Existing deconvolution techniques either propose generative models which are compatible with different kernels but yield high computational cost, or design discriminative models that are efficient to run but are only suitable for a single kernel. In this chapter, we aim to develop a single model that can handle different types of kernels and different noise levels. Such a problem is defined as general non-blind deconvolution. Specifically, we employ the residual learning strategy and propose a very deep convolutional neural network which predicts the residual between a pre-deconvolved image and the corresponding sharp image, rather than only the sharp image. We show that the residual learning strategy can make it easier to train a single model for different kernels and noise, yielding high effectiveness and efficiency. Quantitative evaluations demonstrate the practical applicability of the proposed model on different types of blur kernels, as well as state-of-the-art performance on synthesized blurry images.

5.1 Introduction

The blurry effect in an image is sometimes a nuisance, which is caused by various reasons, such as the relative motion between the scene and the camera during exposure, the failure of focusing all view field, and the atmospheric turbulence in long-distance photographing. The resultant blurry images pose challenges for analysing and understanding the image content in high-level vision tasks. Thus, removing blur is a crucial step in enhancing the quality of images.

Non-blind image deconvolution is a classic low-level vision problem, which assumes that the blur kernel is known and restores the latent sharp image from both the blurry image and the kernel. This problem is ill-posed mainly because of the noise that corrupts the blurry image. The randomness of noise causes the fact that there may be multiple possible sharp images corresponding to the observation, similar to the problem of image denoising. More importantly, even though the noise is independent of the sharp image and the blur kernel, this relationship would be violated when the blurry and noisy image is observed. That is to say, given the degraded image, the sharp image, the blur kernel, and the noise are correlated with each other. If we apply the deconvolution operation (*e.g.* Wiener Deconvolution), the resultant image may contain transformed noise that is highly dependent on the sharp image and the blur kernel. All these aspects make it difficult to estimate the latent sharp image from a single blurry observation even if the blur kernel is known.

A critical issue is that the noise is correlated with the sharp image and the known kernel given the blurry image. This leads to the fact that the generative models have to set different hyper-parameters for different images, and even need to estimate the noise level if it is unknown [140]. Similarly, the discriminative models are generally trained under a specific setting (for example, a certain kernel and a certain noise level), and training multiple models is extremely computational-costly. A model selection step is thus needed when processing a degraded image, limiting the applicability of a single model to images with different degradations. From the practical perspective, it is not feasible to store multiple models in an imaging device with limited memory, or to spend several minutes to produce a high-quality image. Therefore, for the general non-blind

deconvolution purpose, it is required to develop a single model that can handle different blur kernels and different noise levels.

To accomplish the above goal, a direct and effective way is to increase model complexity as much as possible. The great successes obtained by deep convolutional neural networks (CNNs) in various vision tasks have demonstrated the importance of large receptive field and strong nonlinearity in analysing image content. In particular, ResNet with 101 convolutional layers, has shown the state-of-the-art performances in 1000-class ImageNet classification, 200-class ImageNet detection, and 80-class MSCOCO detection/segmentation [64]. InceptionNet [161] and VGG [150] are also very deep architectures and among the top-performed models in the above tasks. All these facts motivate us to develop very deep architectures for general non-blind deconvolution.

To explain the restriction of existing discriminative learning methods [137, 139, 142, 183] for single kernel and single noise level, it should be noted that those models take as input a degraded image and directly regress the ground truth image. We argue that the input degraded samples occupy the space with almost the same size as the output ground truth samples, and there may be many one-to-one mappings between the samples of input and output spaces. The restoration function can be regarded as an approximation to an injective function, which is difficult to train for various cases of degradation. To alleviate this issue, we can either enrich the input space or shrink the output space. The first option is to involve more training samples without changing the output space. For examples, in denoising auto-encoder, random noise is added to the input, generating many corrupted input samples while remaining the output samples clean [169]. Regarding the second option, it has been recently proven that residual learning [64,87,88] makes the training easier, where the residual can be regarded as the ground truth samples located in a reduced-sized space. This will be elaborated in Section 5.3.4.

With the above knowledge in mind, in this chapter, we propose to construct a very deep CNN architecture for the task of general non-blind deconvolution. Specifically, we design the model to predict the residual between a pre-deconvolved image and the corresponding sharp image, by which way the model is responsible for detecting the information that do not belong to the sharp image. The pre-deconvolution is implemented via a regularized wiener de-

convolution method. Such a strategy, on one hand, facilitates the training of the model, and on the other hand, makes it tractable to remove the degradation under different settings (*e.g.* different kernels and different noise levels). Moreover, we try to explain in more details the reason why residual learning can be regarded as an option of reducing the output space and thus facilitate training. Finally, extensive experiments on popular image datasets demonstrate the state-of-the-art performance of the proposed method on different settings of blur degradation. Importantly, our model can achieve very high efficiency through the benefits of the parallel computation on GPU devices.

The rest of the chapter is organised as follows. Section 5.2 gives a review of related work on both non-blind deconvolution and residual learning. The proposed deep CNN model for general non-blind deconvolution, the training process, and an analysis on residual learning are presented in Section 5.3. Extensive experiments are conducted in Section 5.4 to verify the effectiveness of the proposed model. Section 5.5 finally draws the conclusion of this chapter.

5.2 Related Work on Residual Learning

Deep CNN has been shown to be a very powerful architecture in various vision tasks. The very recent progress in designing deep models indicates that residual learning can substantially boost the performance of deep models in large-scaled image classification and object detection tasks [64]. Therefore, the convolutional layers can be stacked to hundreds of or even a thousand layers, while keeping the stacked deep model trainable. This idea shares similar spirits with the recurrent neural network, *i.e.* reusing the signal of previous states. This makes it easier to learn the residual signal than to learn the original, unreferenced signal.

With the idea of residual learning in mind, several attempts on image restoration tasks have been proposed. Kiku et al. [87] proposed the residual interpolation method for the task of colour image demosaicking. In image super-resolution, Kim et al. [88] designed a deep CNN model by stacking the convolutional layers, which predicts the residual between the low-resolution and high-resolution images.

Along with the success of residual learning, the current work is trying to use this strategy to develop a general non-blind deconvolution model, which, to the

Figure 5.1: Illustration of the image residual. (a) The original sharp image x. (b) The blur kernel k. (c) The blur image y. (d) The residual r_b. (e) The pre-deconvolved image $\hat{x}$. (f) The residual r.

best of our knowledge, is the first attempt in such a task. More importantly, we also aim to reveal why residual learning is helpful in different image restoration tasks.

5.3 Deep CNNs for General Non-blind Deconvolution

In this section, we detail the proposed method for general non-blind deconvolution. Before presenting the network architecture, we first give an analysis on what is predicted by our model for the general purpose.

5.3.1 Residual analysis

A general formulation of a blurry image y is expressed as

$$y = k * x + n, \tag{5.1}$$

where x is the corresponding sharp image, k denotes the blur kernel, and n is the additive noise. To set up the residual, we have two options. The one is the residual between the sharp image and the blurry image, meaning that

$$r_b = x - y = x - (k * x + n) = (\delta - k) * x - n, \tag{5.2}$$

where δ is a delta kernel whose center is 1 and the rests are 0. In this case, the residual is nothing but a feature map which exhibits very similar structures to the original sharp image. This can be noticed from Fig. 5.1(d). Thus, predicting r_b based on y is similar to predicting x based on y. It should be noted that for a dense prediction task (*e.g.* image deconvolution), the deep CNN model formulates a mapping between a pixel and its centred local region. If the local region is blurred, the filters of the convolutional layers need to be provided with the inverse kernel information, as in [183]. Apparently, such an operation cannot be generalised to a general deconvolution case, since the CNN model is correlated with the blur kernel.

Another option is to calculate the residual between the sharp image and a pre-deconvolved image. Here we use the direct deconvolution [69, 142] which solves the following objective function to yield a blur inversion:

$$\|y - k * x\|_F^2 + \alpha(\|\nabla_h * x\|_F^2 + \|\nabla_v * x\|_F^2) + \beta\|x\|_F^2, \tag{5.3}$$

where $\|\cdot\|_F$ is the Frobenius norm, ∇_h and ∇_v are the horizontal and vertical gradient operators, respectively, and α and β are regularization parameters. The solution of Eq. (5.3) has a closed form through Fourier expressions, that is,

$$\hat{x} = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(k)}\mathcal{F}(y)}{|\mathcal{F}(k)|^2 + \alpha\mathcal{G} + \beta}\right), \tag{5.4}$$

where $\mathcal{F}$ denotes the Fourier transform, $\overline{\mathcal{F}(\cdot)}$ is the complex conjugate, $\mathcal{G} = |\mathcal{F}(\nabla_h)|^2 + |\mathcal{F}(\nabla_v)|^2$, and the multiplication and the division are element-wise operations. Then, we can compute the residual between $\hat{x}$ and x by substituting y with Eq. (5.1):

$$\begin{aligned} r &= x - \hat{x} \\ &= x - \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(k)}\mathcal{F}(k * x + n)}{|\mathcal{F}(k)|^2 + \alpha\mathcal{G} + \beta}\right) \\ &= \mathcal{F}^{-1}\left(\frac{\alpha\mathcal{G} + \beta}{A}\mathcal{F}(x) - \frac{\overline{\mathcal{F}(k)}}{A}\mathcal{F}(n)\right), \end{aligned} \tag{5.5}$$

where $A = |\mathcal{F}(k)|^2 + \alpha\mathcal{G} + \beta$. From Eq. (5.5), it can be seen that if $\alpha = 0$ and $\beta = 0$, the residual is the deconvolved noise, *i.e.* $r = k_- * n$, where k_- is the inverse kernel of k. If $\alpha \neq 0$ or $\beta \neq 0$, the residual contains the deconvolved noise as well as a corrupt signal of x. In either case, we note that r is correlated with n and k. For the general deconvolution purpose, our aim is to predict a signal that is as less correlated with the image x as possible. In other words, the residual contains no signal from x in an ideal case. Regarding this, we illustrate the residual images r in Fig. 5.1(f), from which we do not see much information belonging to x. Comparing the figures between Figs. 5.1(e) and 5.1(f), we also note that r and $\hat{x}$ have a well-aligned relationship between a pixel in r and its

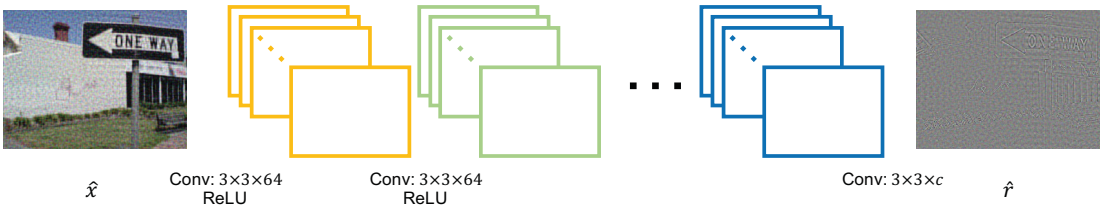


Figure 5.2: The network architecture. The model takes a pre-deconvolved image as input and estimates the residual image. $c = 3$ for colour images and $c = 1$ for grey images.

centred local region in $\hat{x}$. In addition, most of the signals in the residual images can be regarded as noise, while very limited information of edges and textures belonging to x are exhibited. All these facts inform us that it may be easier to train a CNN model to predict the residual in Eq. (5.5) than to predict x directly.

5.3.2 Network architecture

As analysed in last subsection, the proposed model takes the pre-deconvolved image $\hat{x}$ in Eq. (5.4) as input. To design the body of the network, we are inspired by the existing deep models, specifically, the VGG models [150]. That is, the network is a stack of N convolutional layers, but here, we remove all pooling layers as in [88], because pooling may cause information loss and is harmful to restoration tasks. The number of kernels for all layers except the last one is set to 64, and that for the last one is according to the type of the input image, *i.e.* 3 for colour images and 1 for grey images. In each convolutional layer, the kernel spatial size is 3×3. Compared with a larger kernel size, such a design can increase both the strength of model nonlinearity and the effective receptive field size, while the number of model parameters does not grow vastly. For nonlinear activation, a ReLU layer follows each convolutional layer except for the last convolutional layer. The network architecture is depicted in Fig. 5.2.

The effective receptive field size of our CNN model is $(2N + 1) \times (2N + 1)$. Note that the pre-deconvolution in Eq. (5.4) can amplify the noise, which is observed from Figs. 5.1(e) and 5.1(f). Hence, a large receptive field size is required to calculate the residual from a large local region such that sufficient

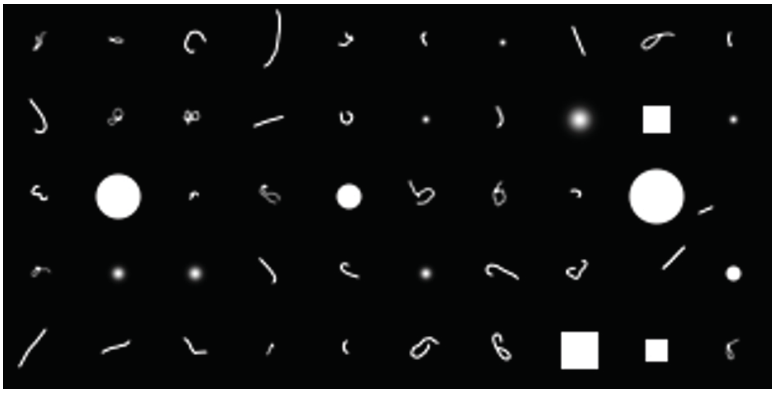


Figure 5.3: Examples of training kernels.

structural information can be involved. This has also been noted by the previous work [142]. The performances for different sizes will be investigated in Section 5.4.2.

We notice that either the method in [142] or the method in [183] produces an output whose size is smaller than the input. To make full use of the input data and the computation, the boundary pixels of the image will be also predicted. This is accomplished by padding the intermediate feature maps with zeros such that the spatial sizes of the feature maps and the output residual are the same as the input image.

Once the residual image $\hat{r}$ is predicted given the input pre-deconvolved image $\hat{x}$, the final result of deconvolution is computed via $x^* = \hat{x} + \hat{r}$.

5.3.3 Training

Similar to the existing discriminative learning methods [139, 142], our proposed model is trained with the Euclidean distance as the training objective:

$$\ell(\hat{x}, r) = \| f_\Theta(\hat{x}) - r \|_F^2, \tag{5.6}$$

where f_Θ specifies the model function, and Θ is the set of all kernel parameters. The parameters Θ are updated by minimising Eq. (5.6) using the back-propagation algorithm [135].

The training dataset is a set of 30000 images randomly sampled from the

Table 5.1: Training parameters.

Parameter	Value
batch size	64
optimization algorithm	SGD
initial learning rate	0.1
learning rate update policy	step
decreasing factor of learning rate	0.1
step size for decreasing learning rate	50000
momentum	0.9
weight decay	0.0001
clip gradients	1
max number of iterations	200000
learning rate multiplier for filter parameters	1
learning rate multiplier for bias terms	0.1

validation set of COCO [107]. Recall that our purpose is to train a general non-blind deconvolution model. The training images should contain different types of blur such that the model has a strong capability of handling different degradations. For this, we collect four types of blur kernels, which are:

- Motion kernels that characterise motion blur. We synthesise these kernels by using the method in [12], where the kernel size is set to 31×31 and the motion parameters (*e.g.* amount of shakes, maximum length of trajectory, and exposure time) are set to diversify the motion trajectory. 100 blur kernels are finally generated.

- Gaussian kernels that characterise atmospheric turbulence blur. The standard derivation of Gaussian uniformly ranges from 0.5 to 3.2 with the step of 0.1, resulting in 28 kernels.

- Circle kernels that characterise defocus blur. The radius of circle uniformly ranges from 3 to 11 with the step of 1, resulting in 9 kernels.

- Square kernels which we use to further enhance the capacity of the model. The side length of square uniformly ranges from 5 to 25 with the step of 2, resulting in 11 kernels.

All the 148 kernels are of size 31×31 by optionally padding with zeros. Several examples of the kernels are illustrated in Fig. 5.3. To generate a blurry image, the sharp image is convolved with a kernel randomly selected from the 148 kernels. Gaussian noise is then added whose standard derivation is drawn from a uniform distribution between 0.008 and 0.06. Such degrading operations are conducted twice for each image, totally generating 60000 blurry images. Once the blur images are synthesised, we use Eqs. (5.4) and (5.5) to compute the pre-deconvolved images and the residual images, by setting $\alpha = 0.002$ and $\beta = 0.001$. To make the sizes of the input training samples consistent, we divide the images into 128×128-sized sub-images with no overlap.

The training phase as well as the test phase are implemented based on the Caffe package [81]. The experiments are carried out on an DevBox with four Nvidia Titan X GPUs, but note that two GPU devices are used for training where each GPU holds a mini-batch of 32 training samples, thus totally 64 training samples are involved in each update iteration. We summarise the necessary training parameters in Table 5.1 to facilitate the reproduction of model training. As seen, the gradient clipping technique is employed which can avoid gradient explosion when using a large learning rate. To ensure numerical stability in float-type computation, the pixel range of the input pre-deconvolved images is normalised from $[0, 255]$ to $[0, 1]$, and that of the residual images is also normalised accordingly. The maximal iterations for training and the step size for decreasing the learning rate are set empirically. To verify that training has converged under such settings, we plot the curve of performance on a validation set which contains 200 images sampled from the COCO training dataset and degraded by the ways introduced previously, as shown in Fig. 5.4.

A trick that we use in the training phase is a small learning rate for bias terms in convolutional layers. We note that in the designed CNN model, the residual image has close pixel values (*i.e.* intensity) to the input pre-deconvolved image. This indicates that there exist solutions of the model parameters in which the bias terms are close to zero. We observed from experiments that a large learning rate for bias terms made the training loss oscillate seriously. Thus, we set the learning rate multiplier for all bias terms to 0.1, which can perform surprisingly well.

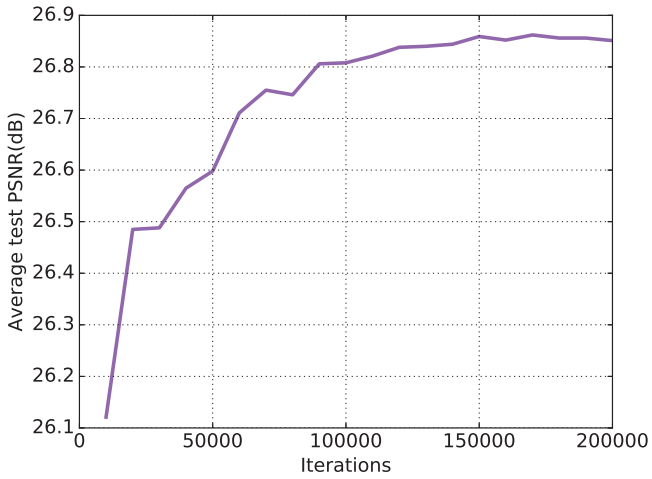


Figure 5.4: The curve of validation performance during training.

5.3.4 Discussion

In this subsection, we aim to explain why the residual learning is so important for image restoration tasks like image non-blind deconvolution.

We begin with the analysis on the property of residual images relative to the original images. We randomly select a degraded image and plot its histogram and the histogram of the corresponding absolute residual image $|r|$, as shown in Fig. 5.5. It can be seen that the original image covers a large range of pixel values while the absolute residual image concentrates its pixels in low values and rarely takes values above 100. This means that the entropy of the histogram of $|r|$ is smaller than that of x, which is further verified through an investigation on the validation image set. As illustrated in Fig. 5.6, the ratio

$$\frac{\text{histogram entropy of } x}{\text{histogram entropy of } |r|} \tag{5.7}$$

is always greater than 1 for all images. Such a phenomenon informs us that *the size of the residual image space is generally smaller and simpler than that of the original image space*. Hence, by replacing the sharp images with the residual images, the output space of the model is shrunk. A condition that may violate such observations is that the degradation is extremely severe, in which case the signal-to-noise ratio is very low and it is almost impossible to restore valuable

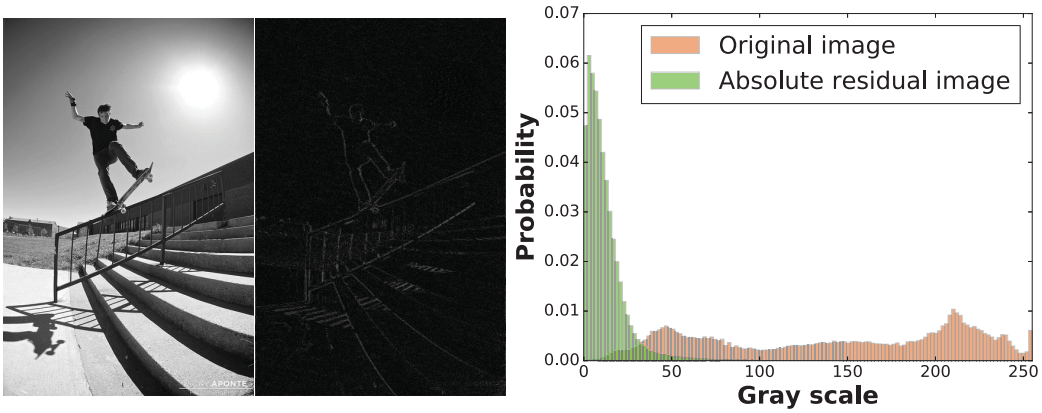


Figure 5.5: Histograms of the original image and the residual image. From left to right: original sharp image, absolute residual image, and histograms.

information of signals from the degraded images.

The next point we claim is that *given the input data unchanged, the required model complexity will decrease along with the reduction of the output space size.* To understand this, we illustrate an extreme case where the input values and the output values have a one-to-one relationship, inducing an injective mapping between them, as shown in Fig. 5.7(a). By compute the residual as the output, we obtain the mapping relationships between multiple input values and an output value. In this way, the output space is shrunk, as illustrated in Fig. 5.7(b). The required model complexity for predicting residual images is lower than that for predicting original sharp images, and thus, residual learning can make it easier to train a CNN model. This is similar to the fact that an optimal classifier for 1000 categories is more complex than an optimal classifier for 100 categories, because more categories require a more complex model.

Residual learning shares similar spirits with the denoising auto-encoder [169] in the sense that they aim to construct the mapping between multiple input values and a single output value. Fig. 5.7(c) illustrates the denoising auto-encoder case where the input space is enriched by adding random noises to input instances, while the output instances are unchanged. The difference between the residual learning and the denoising auto-encoder states that the former shrinks the output space and the later enriches the input space.

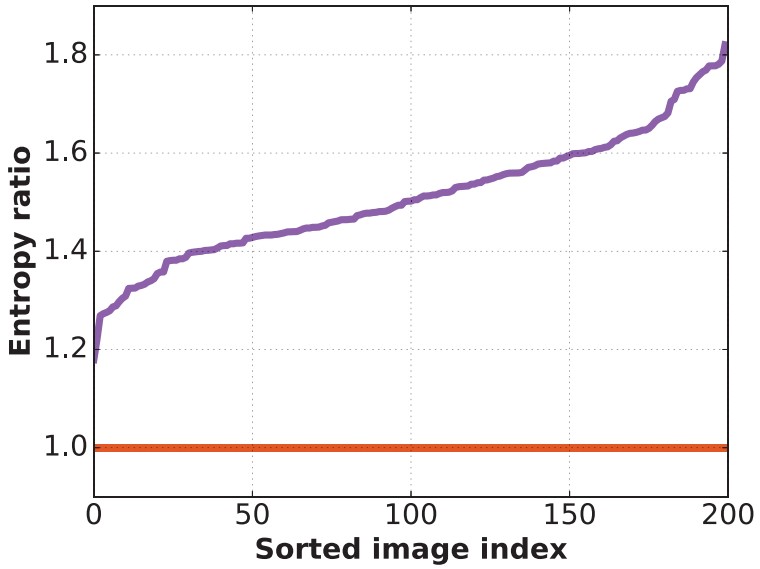


Figure 5.6: The ratio of the histogram entropies between the original images and the corresponding absolute residual images.

5.4 Experiments

In this section, we conduct comprehensive experiments to evaluate the performance of the proposed deep CNN model, which is termed as DEBCNN. The training settings have been detailed in Section 5.3.3. In the following, we consider colour images for evaluation, which means that the proposed model takes a colour image as input and produces the corresponding colour restoration. The model for grayscale images exhibits similar performances and is omitted here.

The test data consist of three datasets: one is from the test set of Berkeley Segmentation Dataset, which has 100 images and is termed as BSD100; one is from the image set used in [142], which has 16 images and is termed as Set16; and the last is from the image set used in [183], which has 30 images and is termed as Set30. Given these datasets, the blurry images are generated by considering the following blur kernels and standard deviations of Gaussian noise:

- The motion kernels employed by Levin et al. [102]. Among the 8 kernels, 4 kernels are used with the noise of $\sigma = 0.01$ (denoted as MotionA) and the others are used with the noise of $\sigma = 0.06$ (denoted as MotionB).

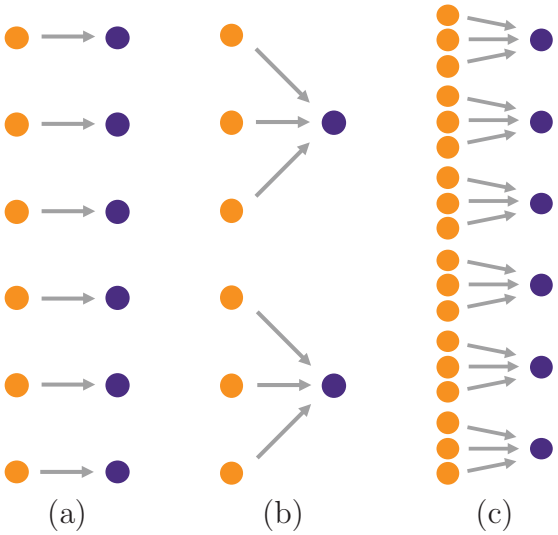


Figure 5.7: Different mappings between the input space (the left of the arrows) and the output space (the right of the arrows). (a) An injective mapping is assumed between the input and output spaces. (b) In the residual learning case, the output space is shrunk by computing the residual. (c) In the denoising auto-encoder case, the input is corrupted by random noises to generate new input samples.

- The Gaussian kernel with the standard derivation of 1.6 and the noise of $\sigma = 0.008$ (denoted as GaussianA).
- The Gaussian kernel with the standard derivation of 3 and the noise of $\sigma = 0.04$ (denoted as GaussianB).
- The Gaussian kernel with the standard derivation of 5 and the noise of $\sigma = 0.04$ (denoted as GaussianC).
- The square kernel with the side size of 19 and the noise of $\sigma = 0.01$ (denoted as SquareA).
- The square kernel with the side size of 13 and the noise of $\sigma = 0.04$ (denoted as SquareB).
- The disk kernel with the radius of 3 and the noise of $\sigma = 0.04$ (denoted as DiskA).

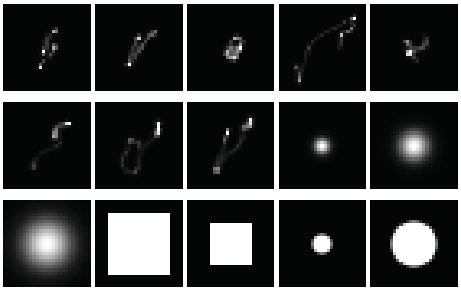


Figure 5.8: Test kernels including 8 motion kernels, 3 Gaussian kernels, 2 square kernels, and 2 disk kernels.

- The disk kernel with the radius of 7 and the noise of $\sigma = 0.01$ (denoted as DiskB).

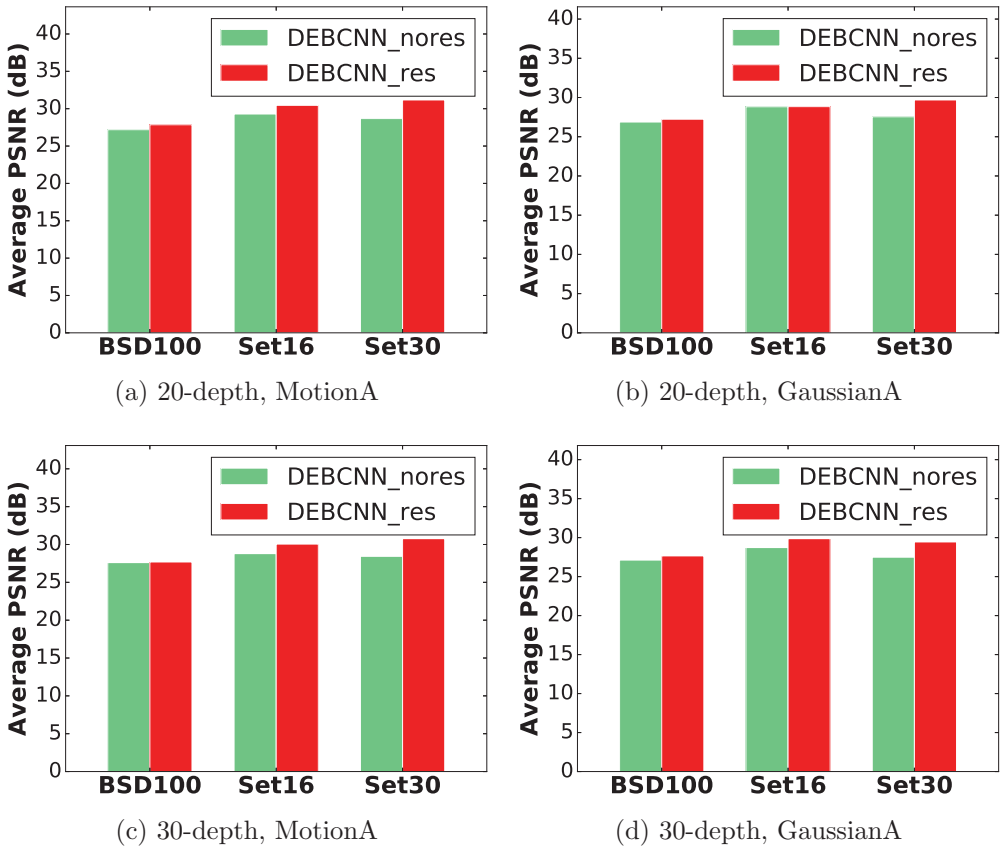
There are totally 15 kernels to be tested, which are illustrated in Fig. 5.8.

To evaluate the qualities of the restored images, we employ the peak signal-to-noise ratio (PSNR) and the structural similarity (SSIM) [175] as assessment metrics.

5.4.1 Effects of residual learning

At first, we conduct experiments to verify the effectiveness of residual learning. We use the terms DEBCNN_res and DEBCNN_nores to specify the models trained with and without the strategy of residual learning, respectively. To be more clear, when residual learning is not applied, the models take the pre-deconvolved image $\hat{x}$ as input, and directly produce x through the convolutional layers, instead of r. The depths of the models that are investigated include 20 and 30. The performances are evaluated on the three test datasets including BSD100, Set16, and Set30. We select two cases from the designed degradations, namely MotionA and GaussianA, for restoration.

Fig. 5.9 illustrates the achieved results, from which we see that in almost all cases, the residual learning can help the models produce better performances. This is consistent with the analysis in Section 5.3.4. Fig. 5.10 displays visual examples. It is observed that in the results by DEBCNN_nores, the distant scenes are not well restored, exhibiting severely blurry effects; but DEBCNN_res performs well and preserves noticeable details. We also notice from Fig. 5.9 that



(a) 20-depth, MotionA

(b) 20-depth, GaussianA

(c) 30-depth, MotionA

(d) 30-depth, GaussianA

Figure 5.9: Comparisons between the models trained with and without residual learning.

compared with the results in Set16 and Set30, the improvement of DEBCNN_res over DEBCNN_nores in BSD100 is limited. This may also occur in the following experiments. We understand this observation as: the scales of images in BSD100 are different from the scales of the training images in the COCO dataset. The scale inconsistency could affect the performance of the trained models because during learning, the filters in the convolutional layers can be sensitive to the scale of local structures in images.

5.4.2 Evaluation on different depths

Experiences inform that a deeper model can always produce a better performance in various vision tasks. Hence, in the task of general non-blind deconvolution,

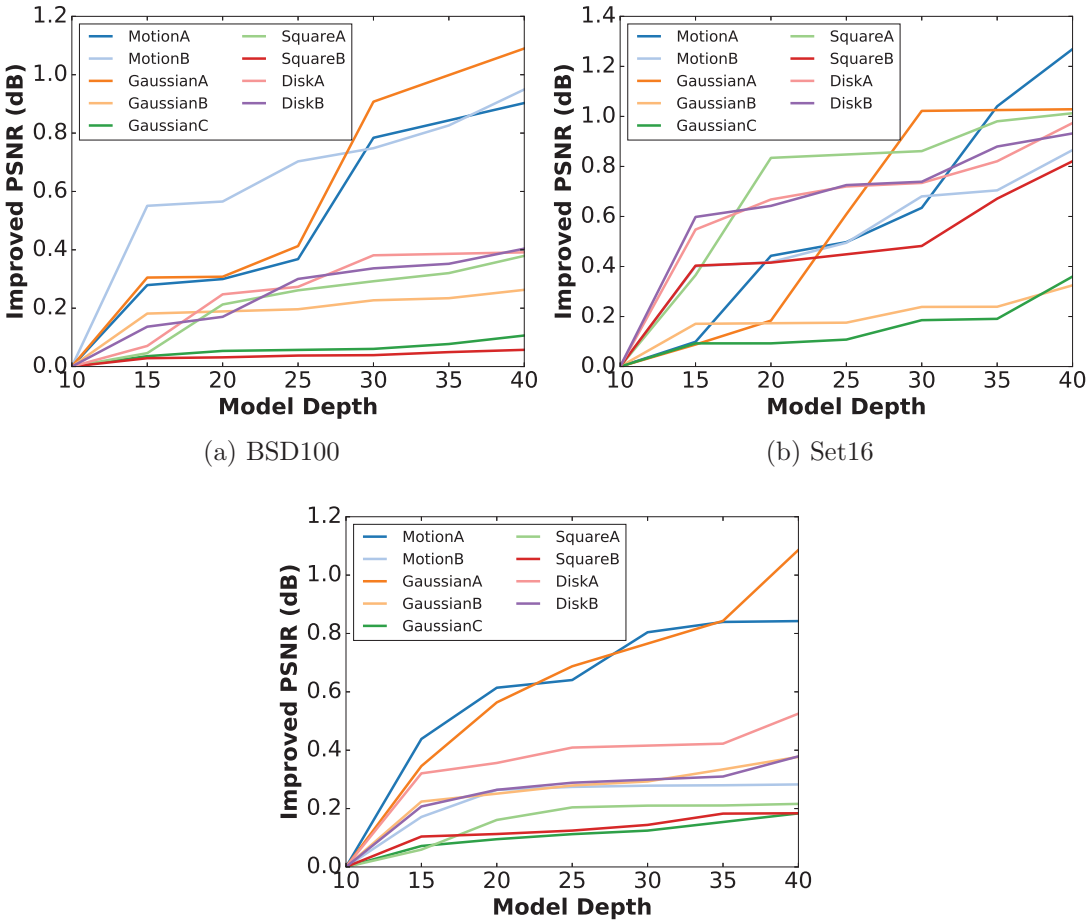
Figure 5.10: Examples restored by DEBCNN_res and DEBCNN_nores. (It is better viewed by zoom-in.)

we investigate the architectures with different model depths to investigate this property. Specifically, we evaluate the models with the depth ranging from 10 to 40 with step size 5. The settings of all convolutional layer (*e.g.* filter size, filter number, and training parameters) are the same as each other. All the degradation cases and the datasets are employed for this investigation.

We denote the performance (here PSNR is used) of the model with depth d as p_d. The 10-depth model is regarded as the basic model. To clearly illustrate the correlation between performance and model depth, we compute the improved performance which is defined as

$$\Delta p_d = p_d - p_{10}. \tag{5.8}$$

Fig. 5.11 presents the obtained results. It is observed that all performance curves increase monotonically with respect to the model depth. This indicates that a deeper model can always produce a better performance, which is consistent with the property stated previously. To explain this, in CNN, a pixel in the output image is related to a local region of the input image with a certain receptive field



(a) BSD100

(b) Set16

(c) Set30

Figure 5.11: Performances of the models with different depths.

size. A deeper model induces a larger receptive field size. When the degradation in an image is serious, a large field of context is needed to estimate the central pixel because a large field has more informative structures than a small field. So the more serious the degradation is, the deeper model is needed. Here, in the task of the defined general non-blind deconvolution, the model should be deep enough such that different levels of degradation can be handled.

From the curves, we also notice that large improvements are achieved for the mild degradation cases, *e.g.* MotionA and GaussianA. By contrary, limited improvements are obtained for the severe degradation cases, *e.g.* GaussianC and SquareB. This may be explained by that textures are required to be synthesised in the severe degradation cases to produce better performances, but in the current work, the training loss (*i.e.* Euclidean distance in Eq. 5.6) is not appropriate for texture synthesis [99].

5.4.3 Comparison to the state-of-the-art

To verify the effectiveness of the proposed model, in this section, we conduct an comparative study with the existing non-blind image deconvolution methods including sparse prior deconvolution [101], hyper-Laplacian prior method [91], expected patch log likelihood (EPLL) [201], iterative decoupled deblurring BM3D (IDD-BM3D) [32], non-locally centralized sparse representation (NCSR) [40], multi-layer perceptron (MLP) [142], and image deconvolution CNN (DCNN) [183]. The proposed model is denoted as DEBCNN in the following, where the selected depth is 40.

All the degradations and the datasets are considered in this comparison. Note that MLP and DCNN are the models trained under specific settings. For example, the MLP models provided in [142] are only for the motion blur, the Gaussian blur, and the square blur. The DCNN model provided in [183] is only for the disk blur. Hence, we evaluate MLP for the cases of MotionA(B), GaussianA(B,C), and SquareA(B), and evaluate DCNN for the cases of DiskA(B). While the models may not be the optimal for all the selected cases, we could examine through such a way that how the models perform when the test degradation is different from the training setting.

Table 5.2: Performance comparison on BSD100. Average PSNR/SSIM are provided with the best value marked in bold. "-" indicates that the model is not suitable for the corresponding case.

Degradation	Levin *et al.* [101]	Krishnan *et al.* [91]	EPLL [201]	IDD-BM3D [32]	NCSR [40]	MLP [142]	DCNN [183]	DEBCNN
MotionA	26.53dB/0.8404	27.14dB/0.8587	27.32dB/0.8625	27.34dB/0.8649	27.72dB/0.8742	26.73dB/0.8448	-	**27.93dB/0.8795**
MotionB	23.36dB/0.7056	24.89dB/0.7870	24.12dB/0.7657	25.05dB/0.7949	24.23dB/0.7327	24.77dB/0.7726	-	**25.50dB/0.8009**
GaussianA	27.09dB/0.8628	26.97dB/0.8578	27.34dB/0.8655	27.17dB/0.8614	27.52dB/0.8748	27.16dB/0.8645	-	**28.47dB/0.8790**
GaussianB	23.87dB/0.7439	23.85dB/0.7521	23.79dB/0.7514	24.16dB/0.7666	24.03dB/0.7532	24.48dB/0.7766	-	**25.34dB/0.7811**
GaussianC	22.52dB/0.6876	22.50dB/0.7065	22.45dB/0.7082	22.75dB/0.7174	22.69dB/0.7073	22.31dB/0.6752	-	**22.79dB/0.7194**
SquareA	23.25dB/0.7154	23.07dB/0.7291	23.36dB/0.7358	23.58dB/0.7371	**23.92dB/0.7539**	22.81dB/0.6975	-	22.90dB/0.7078
SquareB	22.94dB/0.6957	23.18dB/0.7314	22.97dB/0.7252	23.55dB/0.7486	23.29dB/0.7198	23.52dB/0.7375	-	**24.01dB/0.7564**
DiskA	24.75dB/0.7693	25.67dB/0.8150	25.47dB/0.8077	25.87dB/0.8237	25.50dB/0.8001	-	16.51dB/0.4349	**26.61dB/0.8290**
DiskB	24.37dB/0.7590	24.08dB/0.7609	24.55dB/0.7739	24.66dB/0.7740	**25.12dB/0.7942**	-	24.50dB/0.7775	24.66dB/0.7758

Table 5.3: Performance comparison on Set16. Average PSNR/SSIM are provided with the best value marked in bold. "-" indicates that the model is not suitable for the corresponding case.

Degradation	Levin *et al.* [101]	Krishnan *et al.* [91]	EPLL [201]	IDD-BM3D [32]	NCSR [40]	MLP [142]	DCNN [183]	DEBCNN
MotionA	27.53dB/0.8808	30.41dB/0.9380	31.02dB/0.9446	31.79dB/0.9545	28.98dB/0.9146	30.53dB/0.9400	-	**32.73dB/0.9601**
MotionB	20.41dB/0.6710	26.09dB/0.8526	26.01dB/0.8558	27.29dB/0.8785	22.16dB/0.7054	24.27dB/0.7919	-	**27.86dB/0.8824**
GaussianA	30.28dB/0.9247	30.04dB/0.9184	30.64dB/0.9269	30.85dB/0.9341	30.08dB/0.9217	30.94dB/0.9376	-	**31.34dB/0.9439**
GaussianB	25.04dB/0.8130	25.77dB/0.8442	25.74dB/0.8463	26.37dB/0.8578	24.01dB/0.7627	26.74dB/0.8664	-	**26.93dB/0.8691**
GaussianC	23.95dB/0.8003	23.68dB/0.8098	23.63dB/0.8107	24.23dB/0.8224	22.68dB/0.7246	23.75dB/0.7902	-	**24.62dB/0.8276**
SquareA	25.98dB/0.8550	24.57dB/0.8371	25.27dB/0.8508	26.38dB/0.8700	25.18dB/0.8297	26.73dB/0.8788	-	**27.28dB/0.8816**
SquareB	22.85dB/0.7481	24.64dB/0.8292	24.43dB/0.8264	25.37dB/0.8484	23.08dB/0.7384	25.07dB/0.8307	-	**25.74dB/0.8507**
DiskA	22.54dB/0.7394	27.71dB/0.8809	28.02dB/0.8848	28.46dB/0.8955	25.00dB/0.7942	-	17.87dB/0.6383	**28.77dB/0.8995**
DiskB	27.20dB/0.8696	26.17dB/0.8573	27.24dB/0.8756	28.05dB/0.8902	26.91dB/0.8631	-	26.49dB/0.8622	**28.40dB/0.8930**

Table 5.4: Performance comparison on Set30. Average PSNR/SSIM are provided with the best value marked in bold. "-" indicates that the model is not suitable for the corresponding case.

Degradation	Levin *et al.* [101]	Krishnan *et al.* [91]	EPLL [201]	IDD-BM3D [32]	NCSR [40]	MLP [142]	DCNN [183]	DEBCNN
MotionA	28.14dB/0.8456	30.55dB/0.9394	31.38dB/0.9417	31.42dB/0.9504	30.04dB/0.9239	30.89dB/0.9306	-	**32.00dB/0.9523**
MotionB	21.00dB/0.5938	25.50dB/0.8422	24.85dB/0.8477	26.03dB/0.8733	22.51dB/0.6369	24.46dB/0.7529	-	**27.37dB/0.8939**
GaussianA	29.95dB/0.9348	29.35dB/0.9331	30.25dB/0.9400	30.65dB/0.9482	30.11dB/0.9375	30.78dB/0.9488	-	**30.80dB/0.9489**
GaussianB	24.19dB/0.7859	24.44dB/0.8323	24.32dB/0.8351	24.75dB/0.8471	23.37dB/0.7126	25.39dB/0.8613	-	**25.56dB/0.8653**
GaussianC	22.81dB/0.7678	22.44dB/0.7847	22.28dB/0.7856	22.61dB/0.7948	21.81dB/0.6646	22.64dB/0.7572	-	**23.05dB/0.7979**
SquareA	25.52dB/0.8552	23.82dB/0.8304	24.59dB/0.8472	25.62dB/0.8747	24.95dB/0.8385	26.16dB/0.8822	-	**26.31dB/0.8828**
SquareB	22.72dB/0.7001	23.61dB/0.8146	23.14dB/0.8075	23.94dB/0.8325	22.76dB/0.6884	24.31dB/0.8166	-	**25.28dB/0.8585**
DiskA	22.92dB/0.6853	26.82dB/0.8825	26.87dB/0.8886	27.36dB/0.9011	24.89dB/0.7545	-	16.77dB/0.5600	**28.09dB/0.9121**
DiskB	26.53dB/0.8629	25.10dB/0.8539	26.10dB/0.8735	26.99dB/0.8935	26.38dB/0.8688	-	25.33dB/0.8560	**27.70dB/0.9014**

Tables 5.2, 5.3, and 5.4 present the average performances of the considered methods on BSD100, Set16, and Set30, respectively. We see from the results that our method DEBCNN performs better than the competitors in almost all cases. In the SquareA and DiskB cases (which are used respectively for training MLP and DCNN), DEBCNN is still better than the other discriminative learning methods including MLP and DCNN. By comparing different sub-cases for the same type of blur, *e.g.* Motion(A,B), Gaussian(A,B,C), Square(A,B), and Disk(A,B), we note that the MLP models and the DCNN model can perform poorly when the test setting does not fit the corresponding model. That means, for example, if the model is trained for DiskB, its performance in the DiskA case can be very poor. This is explainable by the principle of discriminative learning. Regarding our model, it is trained on a large amount of images under different degradations, which can be seen as a data augmentation operation. But we should emphasise that such a training process is non-trivial as discussed in Section 5.3.4, and we propose to alleviate the training issues via the residual learning strategy, resulting in a single model for the task of general image deconvolution. We also notice that NCSR performs occasionally better than DEBCNN on BSD 100. However, NCSR requires large computation cost on updating the dictionary and the sparse codes, inducing very low efficiency as observed in the analysis about running time in the following.

Figs. 5.12, 5.13, 5.14, and 5.15 present visual examples in the cases of MotionA, GaussianA, SquareA, and DiskB, respectively. It can be seen from the figures that DEBCNN can always restore pleasing details compared with the other methods.

Running time: We also compare the efficiency of all the considered methods. Table 5.5 gives the average running time of different methods on BSD100. Note that for all methods except DEBCNN, the experimental platform is based on Matlab, Intel Xeon CPU (3.47GHz) and 32GB RAM, whereas DEBCNN is implemented based on Python, Nvidia TitanX GPU. While the computation on GPU is highly parallelised, it can be seen from the table that the conventional methods including Levin *et al.* [101], EPLL [201], IDD-BM3D [32], and NCSR [40], require large amounts of time to process an image. Krishnan *et al.* [91] is efficient owing to the closed-form solution of the designed problem. All the deep model-

Blurry image

Levin *et al.* [101] (26.84dB/0.7520)

Krishnan *et al.* [91] (27.57dB/0.7852)

EPLL [201] (27.57dB/0.7706)

IDD-BM3D [32] (27.71dB/0.7860)

NCSR [40] (28.18dB/0.7983)

MLP [142] (27.13dB/0.7551)

DEBCNN (**28.64dB/0.8314**)

Figure 5.12: Visual examples (MotionA). It is better viewed by zoom-in.

Blurry image

Levin *et al.* [101] (24.36dB/0.8271)

Krishnan *et al.* [91] (24.00dB/0.8152)

EPLL [201] (24.48dB/0.8264)

IDD-BM3D [32] (24.63dB/0.8373)

NCSR [40] (24.79dB/0.8516)

MLP [142] (24.60dB/0.8449)

DEBCNN (**25.64dB/0.8790**)

Figure 5.13: Visual examples (GaussianA). It is better viewed by zoom-in.

Blurry image

Levin *et al.* [101] (24.69dB/0.8991)

Krishnan *et al.* [91] (22.84dB/0.8871)

EPLL [201] (23.70dB/0.9039)

IDD-BM3D [32] (24.52dB/0.9184)

NCSR [40] (23.55dB/0.8758)

MLP [142] (25.25dB/0.9123)

DEBCNN (**25.59dB/0.9124**)

Figure 5.14: Visual examples (SquareA). It is better viewed by zoom-in.

Blurry image

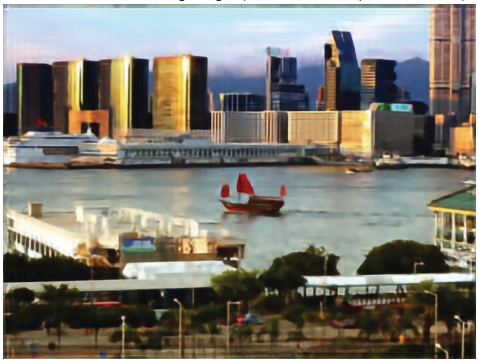
Levin *et al.* [101] (21.99dB/0.7425)

Krishnan *et al.* [91] (20.61dB/0.6863)

EPLL [201] (21.35dB/0.7174)

IDD-BM3D [32] (22.18dB/0.7572)

NCSR [40] (22.12dB/0.7416)



DCNN [183] (21.13dB/0.7017dB)

DEBCNN (**23.24dB/0.7960**)

Figure 5.15: Visual examples (DiskB). It is better viewed by zoom-in.

Table 5.5: Comparison of running time between different methods.

Method	Time
Levin *et al.* [101]	48.33s
Krishnan *et al.* [91]	1.11s
EPLL [201]	882.84s
IDD-BM3D [32]	441.30s
NCSR [40]	1069.86s
MLP [142]	3.45s
DCNN [183]	8.79s
DEBCNN	0.57s

based methods including MLP [142], DCNN [183], and DEBCNN perform very efficiently, and can satisfy real-time requirements by using GPU implementations.

5.5 Conclusion

Most of the existing non-blind deconvolution methods present the models that are specific for blur kernels. They require either high computational consumption or high memory. In this chapter, we are interested in developing a model for general non-blind deconvolution, which means that only one model is needed to efficiently perform deconvolution for different blur kernels. Specifically, we take advantage of CNNs that possess high representational capacity and high efficiency during inference. Different from the conventional CNN model for deconvolution, we employ the residual learning strategy by which the designed model estimates the residuals between a pre-deconvolved image and the corresponding sharp images. We also give an analysis on why the training of CNN models can benefit from residual learning, and how residual learning relates to the denoising auto-encoder. The experimental results demonstrate the effectiveness of residual learning, and also indicate that a deeper model can always produce a higher performance. The comparative studies illustrate that the proposed model achieves superior results over the selected competitors, and has very high efficiency owing to the parallel implementation on GPU devices.

Chapter 6

Conclusions

Image quality enhancement is a critical module in many imaging systems and image-related vision systems, yet it is a challenging task due to the ill-posed nature of the problem. While we would like to remove the external degradations that corrupt an image, it is also important to recover vivid textures which make the image look more natural. As noted, noise, down-sampling operations, and blur cause the details of images to be lost, so information loss is the most important and difficult issue. A common option to handle this is to incorporate prior knowledge about high-quality images which are either obtained according to intuitions or extracted from redundant high-quality image data. A large body of research has been published in recent years on both fashions of getting the prior knowledge. Benefiting from the development of advanced parallel computing techniques (*e.g.* GPU), the deep learning framework has received great attention and success, and has been proven to have a very powerful capability to handle complex vision tasks, including image processing. Hence, in this thesis, we investigate the deep learning framework and propose a series of novel models for the task of image quality enhancement.

Specifically, the first stage of our work examines how the degradation existing in images affects the deep neural networks. In details, we investigate how the signal propagation through a neural network is affected when similar inputs are corrupted by random noise. It is generally expected that similar inputs induce similar signals. However, due to the randomness of the degradation, we find that the noisy effects, which are defined as turbulence, can be strengthened as the

signals are propagated into deep layers. This is contrary to our expectation. To address this issue, we then develop the (stacked) non-local auto-encoder which explicitly exploits self-similarity information in natural images to enhance the stability of signal propagation in neural networks. The self-similarity property implies that there exist many similar patches in images. Thus, we can constrain the corrupted (*e.g.* noisy) version of similar patches and induce similar signals when they are fed into a neural network. This is achieved by adding a similarity regularisation term into the training objective function. Furthermore, a collaborative stabilisation step is designed to further rectify forward propagation. The experimental results on image denoising and super-resolution indicate that the performance of the designed model is superior to its competitors.

The second stage of our work concerns the effects of receptive field size and model depth with respect to the image restoration performance. A general property of deep models states that a deeper model has a larger receptive field size and higher representation capacity, and produces better performance. While this is true for the image restoration task, we are interested in discovering whether it is possible to use a shallower model but with a large receptive field size to produce pleasing results. In this work, we focus on CNN models in the single image super-resolution (SISR) task. For this investigation, we design a strategy by using the dilated convolution and the 1×1 convolution, that can evaluate the effects of one factor by fixing another. In particular, we can employ the dilation convolution to enlarge the receptive field size while retaining the shallow model; and we can also stack 1×1 convolution layers to increase the model depth while keeping the receptive field size the same. Our findings from the comprehensive experiments show that the SISR task is more sensitive to receptive field size than model depth. It is also noticed that good performance can be achieved only when the model depth is congruent with the receptive field size. These empirical findings suggest that it is possible to use a shallower architecture to achieve a performance comparable to a deeper architecture. This is helpful for designing effective CNN models which can save both computational and memory cost.

The final stage of our work develops a single discriminative model for the general non-blind deconvolution task, where "general" means that the model can address different types of blur degradation. The image deconvolution techniques

can be grouped into generative models and discriminative models. The generative models rely on an objective function that is applicable for different blur kernels, but they require a complex optimisation process, inducing low efficiency. The discriminative models only need a forward process for restoration, having high efficiency, but they are usually trained for a single blur setting. For general purposes, we propose a residual learning strategy. Specifically, given a blurry image, a deep convolutional neural network is designed to predict the residual between the sharp image and the pre-deconvolved image by direct deconvolution. We empirically and analytically show that the residual learning strategy makes the training of the restoration model easier, and also enhances the reliability of the trained model for handling different blur kernels and different noise levels. Extensive experiments indicate that the proposed model produces state-of-the-art performance on a large number of blurry images, and exhibits very high efficiency by using parallel computation on GPU devices.

References

[1] F. Agostinelli, M. R. Anderson, and H. Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *NIPS*, 2013, pp. 1493–1501. 17, 45

[2] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006. 24

[3] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *NIPS*, 2014, pp. 2654–2662. 66

[4] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos, "Bayesian blind deconvolution with general sparse image priors," in *ECCV*, 2012, pp. 341–355. 23

[5] D. H. Ballard, "Modular learning in neural networks." in *Proceedings of AAAI*, 1987, pp. 279–284. 25, 26

[6] Y. Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009. 9, 33, 34

[7] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle *et al.*, "Greedy layer-wise training of deep networks," *NIPS*, vol. 19, p. 153, 2007. 40

[8] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975. 47

[9] M. Bertero and P. Boccacci, *Introduction to Inverse Problems in Imaging*. CRC press, 1998. 22

[10] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," in *BMVC*, 2012. 57, 70

[11] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. A. Morel, "Neighbor embedding based single-image super-resolution using semi-nonnegative matrix factorization," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2012, pp. 1289–1292. 20

[12] G. Boracchi and A. Foi, "Modeling the performance of image restoration from motion blur," *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3502–3517, 2012. 108

[13] Y.-l. Boureau, Y. L. Cun *et al.*, "Sparse feature learning for deep belief networks," in *NIPS*, 2008, pp. 1185–1192. 33, 35, 44

[14] J. Bruna, P. Sprechmann, and Y. LeCun, "Super-resolution with deep convolutional sufficient statistics," *ICLR*, 2016. 21, 66

[15] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2005, pp. 60–65. 15, 33, 38, 39, 45, 47, 52, 53, 58

[16] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with bm3d?" in *CVPR*, 2012, pp. 2392–2399. 17, 32, 39, 45, 46, 48, 51, 52, 53, 58

[17] H. C. Burger, C. Schuler, and S. Harmeling, "Learning how to combine internal and external denoising methods," in *GCPR*, vol. 8142, 2013, p. 121. 17, 45

[18] J.-F. Cai, S. Osher, and Z. Shen, "Split bregman methods and frame based image restoration," *Multiscale modeling & simulation*, vol. 8, no. 2, pp. 337–369, 2009. 2

[19] A. Chakrabarti, T. Zickler, and W. T. Freeman, "Analyzing spatially-varying blur," in *CVPR*. IEEE, 2010, pp. 2512–2519. 23

[20] T.-M. Chan, J. Zhang, J. Pu, and H. Huang, "Neighbor embedding based super-resolution algorithm through edge detection and feature selection," *Pattern Recognition Letters*, vol. 30, no. 5, pp. 494–502, 2009. 20

[21] P. Chandramouli and A. Rajagopalan, "Inferring image transformation and structure from motion-blurred images," in *Proceedings of British Machine Vision Conference*, 2010, pp. 73.1–73.12. 25

[22] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2004, pp. I–I. 20

[23] O. Chapelle, B. Schlkopf, and A. Zien, *Semi-Supervised Learning*. The MIT Press, 2010. 42

[24] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," *ICLR*, 2015. 69

[25] ——, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *arXiv preprint arXiv:1606.00915*, 2016. 69, 73

[26] Y. Chen, R. Ranftl, and T. Pock, "Insights into analysis operator learning: From patch-based sparse models to higher order mrfs," *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 1060–1072, 2014. 33

[27] S. Cho, H. Cho, Y.-W. Tai, and S. Lee, "Registration based non-uniform motion deblurring," *Computer Graphics Forum*, vol. 31, no. 7, pp. 2183–2192, 2012. 25

[28] D. D. Cox and R. L. Savoy, "Functional magnetic resonance imaging (fmri)'brain reading': detecting and classifying distributed patterns of fmri activity in human visual cortex," *Neuroimage*, vol. 19, no. 2, pp. 261–270, 2003. 32, 33

[29] Z. Cui, H. Chang, S. Shan, B. Zhong, and X. Chen, "Deep network cascade for image super-resolution," in *ECCV*, 2014, pp. 49–64. 21, 34, 45

[30] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on image processing*, vol. 16, no. 8, pp. 2080–2095, 2007. 16, 17, 33, 46, 52, 53, 58

[31] A. Danielyan, V. Katkovnik, and K. Egiazarian, "Image deblurring by augmented lagrangian with bm3d frame prior," in *Workshop on Information Theoretic Methods in Science and Engineering*, 2010, pp. 16–18. 24

[32] ——, "Bm3d frames and variational image deblurring," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1715–1728, 2012. 118, 119, 120, 121, 122, 123, 124, 125, 126

[33] M. Delbracio, P. Musé, A. Almansa, and J.-M. Morel, "The non-parametric sub-pixel local point spread function estimation is a well posed problem," *International Journal of Computer Vision*, vol. 96, no. 2, pp. 175–194, 2012. 5

[34] C.-A. Deledalle, L. Denis, and F. Tupin, "Iterative weighted maximum likelihood denoising with probabilistic patch-based weights," *IEEE Transactions on Image Processing*, vol. 18, no. 12, pp. 2661–2672, 2009. 9

[35] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *ECCV*, 2014, pp. 184–199. 20, 45, 54, 58, 59, 63

[36] ——, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016. 20, 70, 73, 94

[37] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *ECCV*. Springer, 2016, pp. 391–407. 21, 85, 94

[38] W. Dong, X. Li, L. Zhang, and G. Shi, "Sparsity-based image denoising via dictionary learning and structural clustering," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 457–464. 17

[39] W. Dong, L. Zhang, and G. Shi, "Centralized sparse representation for image restoration," in *Proceedings of IEEE International Conference on Computer Vision*, 2011, pp. 1259–1266. 24

[40] W. Dong, L. Zhang, G. Shi, and X. Li, "Nonlocally centralized sparse representation for image restoration," *IEEE Transactions on Image Processing*, vol. 22, no. 4, pp. 1620–1630, 2013. 9, 24, 33, 38, 46, 52, 53, 54, 58, 59, 63, 118, 119, 120, 121, 122, 123, 124, 125, 126

[41] W. Dong, L. Zhang, G. Shi, and X. Wu, "Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization," *IEEE Transactions on Image Processing*, vol. 20, no. 7, pp. 1838–1857, 2011. 20, 24

[42] D. L. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613–627, 1995. 16

[43] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995. 16

[44] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011. 48

[45] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006. 17, 52, 53, 58

[46] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *The Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010. 41

[47] W. Fan and D.-Y. Yeung, "Image hallucination using neighbor embedding over visual primitive manifolds," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–7. 19

[48] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," in *Proceedings of ACM SIGGRAPH*, 2006, pp. 787–794. 9, 23

[49] W. T. Freeman, T. R. Jones, and E. C. Pasztor, "Example-based super-resolution," *IEEE Computer Graphics and Applications*, vol. 22, no. 2, pp. 56–65, 2002. 19

[50] W. T. Freeman and E. Pasztor, "Markov networks for low-level vision," *Mitsubishi Electric Research Laboratory Technical Report TR99-08*, 1999. 19

[51] W. T. Freeman and E. C. Pasztor, "Learning to estimate scenes from images," *Proceedings of Advances in Neural Information Processing Systems*, pp. 775–781, 1999. 19

[52] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, "Learning low-level vision," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000. 19

[53] X. Gao, K. Zhang, D. Tao, and X. Li, "Image super-resolution with sparse neighbor embedding," *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3194–3205, 2012. 20

[54] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 721–741, 1984. 2

[55] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. 1, 14, 16

[56] B. Goossens, Q. Luong, A. Pizurica, and W. Philips, "An improved non-local denoising algorithm," in *International Workshop on Local and Non-Local Approximation in Image Processing*, 2008, pp. 143–156. 15

[57] R. Goroshin and Y. LeCun, "Saturating auto-encoders," *arXiv preprint arXiv:1301.3577*, 2013. 45

[58] M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust l1 optimal camera paths," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 225–232. 1

[59] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2862–2869. 15, 33

[60] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, "Convolutional sparse coding for image super-resolution," in *ICCV*, 2015, pp. 1823–1831. 45

[61] K. Guo, X. Yang, R. Zhang, and S. Yu, "Learning super resolution with global and local constraints," in *Proceedings of IEEE International Conference on Multimedia and Expo*. IEEE, 2009, pp. 590–593. 18

[62] A. Gupta, N. Joshi, C. L. Zitnick, M. Cohen, and B. Curless, "Single image deblurring using motion density functions," in *ECCV*, 2010, pp. 171–184. 25

[63] S. Harmeling, H. Michael, and B. Schölkopf, "Space-variant single-image blind deconvolution for removing camera shake," in *Proceedings of Advances in Neural Information Processing Systems*, 2010, pp. 829–837. 25

[64] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CVPR*, 2016. 29, 66, 72, 101, 102

[65] L. He, H. Qi, and R. Zaretzki, "Beta process joint dictionary learning for coupled feature spaces with application to single image super-resolution," in *CVPR*, 2013, pp. 345–352. 20

[66] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006. 40

[67] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. 26, 33, 35, 39, 44

[68] G. E. Hinton and R. S. Zemel, "Autoencoders, minimum description length, and helmholtz free energy," *NIPS*, pp. 3–3, 1994. 44

[69] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Schölkopf, "Fast removal of non-uniform camera shake," in *ICCV*. IEEE, 2011, pp. 463–470. 24, 105

[70] M. Hirsch, S. Sra, B. Schölkopf, and S. Harmeling, "Efficient filter flow for space-variant multiframe blind deconvolution," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 607–614. 25

[71] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 25

[72] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian, "A real-time algorithm for signal analysis with the help of the wavelet transform," in *Wavelets*. Springer, 1990, pp. 286–297. 67, 68

[73] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. 44

[74] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *CVPR*. IEEE, 2015, pp. 5197–5206. 70

[75] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, no. 1, pp. 106–154, 1962. 25

[76] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015. 29

[77] A. Ivakhnenko, "Cybernetic predicting devices," Tech. Rep. 25

[78] A. G. Ivakhnenko, "The group method of data handling-a rival of the method of stochastic approximation," *Soviet Automatic Control*, vol. 13, no. 3, pp. 43–55, 1968. 25

[79] ——, "Polynomial theory of complex systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 1, no. 4, pp. 364–378, 1971. 25

[80] V. Jain and S. Seung, "Natural image denoising with convolutional networks," in *NIPS*, 2009, pp. 769–776. 17, 45, 46

[81] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACMMM*. ACM, 2014, pp. 675–678. 71, 109

[82] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," *ECCV*, 2016. 21

[83] N. Joshi, W. Matusik, E. H. Adelson, and D. J. Kriegman, "Personal photo enhancement using example images," *ACM Transactions on Graph*, vol. 29, no. 2, p. 12, 2010. 24

[84] M. Jung, X. Bresson, T. F. Chan, and L. A. Vese, "Nonlocal mumford-shah regularizers for color image restoration," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1583–1598, 2011. 23

[85] T. Kadir and M. Brady, "Saliency, scale and image description," *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, 2001. 81

[86] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981. 18

[87] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Residual interpolation for color image demosaicking," in *ICIP*. IEEE, 2013, pp. 2304–2308. 101, 102

[88] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *CVPR*, 2016. 20, 21, 32, 66, 69, 70, 71, 72, 94, 101, 102, 106

[89] ——, "Deeply-recursive convolutional network for image super-resolution," in *CVPR*, 2016. 21, 66

[90] K. I. Kim and Y. Kwon, "Single-image super-resolution using sparse regression and natural image prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1127–1133, 2010. 20

[91] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-laplacian priors," in *NIPS*, 2009, pp. 1033–1041. 118, 119, 120, 121, 122, 123, 124, 125, 126

[92] D. Krishnan, T. Tay, and R. Fergus, "Blind deconvolution using a normalized sparsity measure," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 233–240. 23

[93] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105. 29, 66

[94] D. T. Kuan, A. Sawchuk, T. C. Strand, P. Chavel *et al.*, "Adaptive noise smoothing filter for images with signal-dependent noise," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 2, pp. 165–177, 1985. 43

[95] D. Kundur and D. Hatzinakos, "Blind image deconvolution," *IEEE Signal Processing Magazine*, vol. 13, no. 3, pp. 43–64, 1996. 6

[96] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989. 25

[97] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-

propagation network," in *Proceedings of Advances in Neural Information Processing Systems*, 1990, pp. 396–404. 25

[98] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. 29

[99] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint arXiv:1609.04802*, 2016. 21, 82, 118

[100] T. M. Lehmann, C. Gonner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Transactions on Medical Imaging*, vol. 18, no. 11, pp. 1049–1075, 1999. 2

[101] A. Levin, R. Fergus, F. Durand, and W. T. Freeman, "Image and depth from a conventional camera with a coded aperture," *ACM transactions on graphics*, vol. 26, no. 3, p. 70, 2007. 118, 119, 120, 121, 122, 123, 124, 125, 126

[102] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1964–1971. 23, 112

[103] ——, "Understanding blind deconvolution algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 12, pp. 2354–2367, 2011. 23

[104] B. Li, H. Chang, S. Shan, and X. Chen, "Locality preserving constraints for super-resolution with neighbor embedding," in *Proceedings of International Conference on Image Processing*. IEEE, 2009, pp. 1189–1192. 20

[105] ——, "Low-resolution face recognition via coupled locality preserving mappings," *IEEE Signal Processing Letters*, vol. 17, no. 1, pp. 20–23, 2010. 20

[106] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE transactions on image processing*, vol. 10, no. 10, pp. 1521–1527, 2001. 18

[107] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755. 108

[108] S. Linnainmaa, "The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors," *Master's Thesis (in Finnish), Univ. Helsinki*, pp. 6–7, 1970. 25

[109] ——, "Taylor expansion of the accumulated rounding error," *BIT Numerical Mathematics*, vol. 16, no. 2, pp. 146–160, 1976. 25

[110] D. Liu, Z. Wang, B. Wen, J. Yang, W. Han, and T. S. Huang, "Robust single image super-resolution via deep networks with sparse prior," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3194–3207, 2016. 21

[111] L. B. Lucy, "An iterative technique for the rectification of observed distributions," *The Astronomical Journal*, vol. 79, pp. 745–754, 1974. 22

[112] L. Ma, L. Moisan, J. Yu, and T. Zeng, "A dictionary learning approach for poisson image deblurring," *IEEE Transactions on Medical Imaging*, vol. 32, no. 7, pp. 1277–1289, 2013. 24

[113] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Proceedings of IEEE International Conference on Computer Vision*. IEEE, 2009, pp. 2272–2279. 17, 33, 46, 52, 53, 58

[114] S. Mallat, *A wavelet tour of signal processing*. Academic press, 1999. 16, 67, 68

[115] D. J. Mannion, D. J. Kersten, and C. A. Olman, "Regions of mid-level human visual cortex sensitive to the global coherence of local image patches," *Journal of Cognitive Neuroscience*, vol. 26, no. 8, pp. 1764–1774, 2014. 52

[116] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *ICCV*, vol. 2, 2001, pp. 416–423. 47, 70

[117] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1150–1163, 2006. 1

[118] V. Nair and G. E. Hinton, "3d object recognition with deep belief nets," in *NIPS*, 2009, pp. 1339–1347. 40

[119] ——, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of International Conference on Machine Learning*, 2010, pp. 807–814. 28

[120] J. Nakamura, *Image sensors and signal processing for digital still cameras*. CRC press, 2016. 1

[121] K. Nasrollahi and T. B. Moeslund, "Super-resolution: a comprehensive survey," *Machine vision and applications*, vol. 25, no. 6, pp. 1423–1468, 2014. 18

[122] J. Ni, P. Turaga, V. M. Patel, and R. Chellappa, "Example-driven manifold priors for image deconvolution," *IEEE Transactions on Image Processing*, vol. 20, no. 11, pp. 3086–3096, 2011. 23, 24

[123] K. S. Ni and T. Q. Nguyen, "Image superresolution using support vector regression," *IEEE Transactions on Image Processing*, vol. 16, no. 6, pp. 1596–1610, 2007. 20

[124] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990. 14

[125] D. Perrone, A. Ravichandran, R. Vidal, and P. Favaro, "Image priors for image deblurring with uncertain blur," in *Proceedings of British Machine Vision Conference*, 2012, pp. 114.1–114.11. 24

[126] G. Peyré, "Manifold models for signals and images," *Computer Vision and Image Understanding*, vol. 113, no. 2, pp. 249–260, 2009. 23

[127] S. Pinker and A. Prince, "On language and connectionism: Analysis of a parallel distributed processing model of language acquisition," *Cognition*, vol. 28, no. 1, pp. 73–193, 1988. 25

[128] J. B. Pollack, "Recursive distributed representations," *Artificial Intelligence*, vol. 46, no. 1, pp. 77–105, 1990. 26

[129] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Transactions on Image processing*, vol. 12, no. 11, pp. 1338–1351, 2003. 6, 16

[130] J. A. Richards and J. Richards, *Remote sensing digital image analysis*. Springer, 1999, vol. 3. 2

[131] W. H. Richardson, "Bayesian-based iterative method of image restoration," *Journal of the Optical Society of America*, vol. 62, no. 1, pp. 55–59, 1972. 22

[132] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *ICML*, 2011, pp. 833–840. 44, 45

[133] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2005, pp. 860–867. 18

[134] L. I. Rudin and S. Osher, "Total variation based image restoration with free local constraints," in *Proceedings of International Conference on Image Processing*, vol. 1. IEEE, 1994, pp. 31–35. 14

[135] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive Modeling*, vol. 5, no. 3, p. 1, 1988. 107

[136] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015. 9

[137] U. Schmidt, J. Jancsary, S. Nowozin, S. Roth, and C. Rother, "Cascades of regression tree fields for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 677–689, 2016. 101

[138] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *CVPR*, 2014, pp. 2774–2781. 33

[139] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth, "Discriminative non-blind deblurring," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 604–611. 24, 101, 107

[140] U. Schmidt, K. Schelten, and S. Roth, "Bayesian deblurring with integrated noise estimation," in *CVPR*. IEEE, 2011, pp. 2625–2632. 100

[141] B. Schölkopf and C. J. Burges, *Advances in kernel methods: support vector learning*. MIT press, 1999. 26

[142] C. J. Schuler, H. C. Burger, S. Harmeling, and B. Schölkopf, "A machine learning approach for non-blind image deconvolution," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1067–1074. 24, 101, 105, 107, 112, 118, 119, 120, 122, 123, 124, 126

[143] S. Schulter, C. Leistner, and H. Bischof, "Fast and accurate image upscaling with super-resolution forests," in *CVPR*, 2015, pp. 3791–3799. 70

[144] T. Serre, G. Kreiman, M. Kouh, C. Cadieu, U. Knoblich, and T. Poggio, "A quantitative theory of immediate visual recognition," *Progress in Brain Research*, vol. 165, pp. 33–56, 2007. 33

[145] E. Shaked and O. Michailovich, "Iterative shrinkage approach to restoration of optical imagery," *IEEE Transactions on Image Processing*, vol. 20, no. 2, pp. 405–416, 2011. 23

[146] L. Shao, H. Zhang, and G. De Haan, "An overview and performance evaluation of classification-based least squares trained filters," *IEEE Transactions on Image Processing*, vol. 17, no. 10, pp. 1772–1782, 2008. 14

[147] Z. Shen, K.-C. Toh, and S. Yun, "An accelerated proximal gradient algorithm for frame-based image restoration via the balanced approach," *SIAM Journal on Imaging Sciences*, vol. 4, no. 2, pp. 573–596, 2011. 24

[148] M. J. Shensa, "The discrete wavelet transform: wedding the a trous and Mallat algorithms," *IEEE Transactions on signal processing*, vol. 40, no. 10, pp. 2464–2482, 1992. 67

[149] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *CVPR*, 2016, pp. 1874–1883. 21, 66, 85

[150] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 29, 66, 72, 101, 106

[151] S. M. Smith and J. M. Brady, "Susan: a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997. 14

[152] C. K. Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, "Amortised map inference for image super-resolution," *arXiv preprint arXiv:1610.04490*, 2016. 21

[153] A. J. Storkey, "Dynamic structure super-resolution," in *Proceedings of Advances in Neural Information Processing Systems*, 2002, pp. 1295–1302. 20

[154] J. Sun, Z. Xu, and H.-Y. Shum, "Image super-resolution using gradient profile prior," in *CVPR*. IEEE, 2008, pp. 1–8. 18

[155] ——, "Gradient profile prior and its applications in image super-resolution and enhancement," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1529–1542, 2011. 18

[156] J. Sun, N.-N. Zheng, H. Tao, and H.-Y. Shum, "Image hallucination with primal sketch priors," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2003, pp. II–729. 19

[157] L. Sun, S. Cho, J. Wang, and J. Hays, "Edge-based blur kernel estimation using patch priors," in *IEEE International Conference on Computational Photography*, 2013, pp. 1–8. 24

[158] ——, "Good image priors for non-blind deconvolution," in *ECCV*. Springer, 2014, pp. 231–246. 23

[159] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *arXiv preprint arXiv:1602.07261*, 2016. 29

[160] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9. 29

[161] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *arXiv preprint arXiv:1512.00567*, 2015. 29, 66, 101

[162] Y.-W. Tai, N. Kong, S. Lin, and S. Y. Shin, "Coded exposure imaging for projective motion deblurring," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2408–2415. 25

[163] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Transactions on image processing*, vol. 16, no. 2, pp. 349–366, 2007. 14

[164] R. Timofte, V. De, and L. Van Gool, "Anchored neighborhood regression for fast example-based super-resolution," in *ICCV*, 2013, pp. 1920–1927. 20, 54, 58, 59, 63

[165] R. Timofte, V. De Smet, and L. Van Gool, "A+: Adjusted anchored neighborhood regression for fast super-resolution," in *ACCV*. Springer, 2014, pp. 111–126. 20

[166] M. E. Tipping and C. Bishop, "Bayesian image super-resolution," *NIPS*, no. 1279-1286, 2002. 7

[167] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013. 26

[168] M. Vetterli and C. Herley, "Wavelets and filter banks: Theory and design," *IEEE transactions on signal processing*, vol. 40, no. 9, pp. 2207–2232, 1992. 68

[169] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010. 33, 35, 42, 44, 45, 48, 51, 73, 101, 111

[170] J. Wang, S. Zhu, and Y. Gong, "Resolution enhancement based on learning the sparse association of image patches," *Pattern Recognition Letters*, vol. 31, no. 1, pp. 1–10, 2010. 20

[171] R. Wang and D. Tao, "Non-local auto-encoder with collaborative stabilization for image restoration," *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2117–2129, 2016. 21, 73

[172] Y. Wang, L. Wang, H. Wang, and P. Li, "End-to-end image super-resolution via deep and shallow convolutional networks," *arXiv preprint arXiv:1607.07680*, 2016. 21, 66

[173] Z. Wang, Y. Yang, Z. Wang, S. Chang, W. Han, J. Yang, and T. Huang, "Self-tuned deep super resolution," in *CVPR Workshops*, 2015, pp. 1–8. 21

[174] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang, "Deep networks for image super-resolution with sparse prior," in *ICCV*, 2015, pp. 370–378. 21, 94

[175] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004. 74, 114

[176] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, no. 4, pp. 339–356, 1988. 25

[177] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce, "Non-uniform deblurring for shaken images," *International Journal of Computer Vision*, vol. 98, no. 2, pp. 168–186, 2012. 25

[178] N. Wiener, *Extrapolation, interpolation, and smoothing of stationary time series*. MIT press Cambridge, 1949, vol. 2. 22

[179] G. Wu, X. Zhu, Q. Wang, and D. Shen, "Image super-resolution by supervised adaption of patchwise self-similarity from high-resolution image," in *Patch-Based Techniques in Medical Imaging*. Springer, 2015, pp. 10–18. 33

[180] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *NIPS*, 2012, pp. 341–349. 17, 39, 40, 45, 46

[181] S. Xie and S. Rahardja, "Alternating direction method for balanced image restoration," *IEEE Transactions on Image Processing*, vol. 21, no. 11, pp. 4557–4567, 2012. 24

[182] Y. Xie, S. Gu, Y. Liu, W. Zuo, W. Zhang, and L. Zhang, "Weighted schatten p-norm minimization for image denoising and background subtraction," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4842–4857, 2016. 15

[183] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *NIPS*, 2014, pp. 1790–1798. 24, 45, 101, 104, 107, 112, 118, 119, 120, 125, 126

[184] L. Xu, S. Zheng, and J. Jia, "Unnatural l_0 sparse representation for natural image deblurring," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1107–1114. 23

[185] C.-Y. Yang, C. Ma, and M.-H. Yang, "Single-image super-resolution: a benchmark," in *ECCV*. Springer, 2014, pp. 372–386. 18

[186] C.-Y. Yang and M.-H. Yang, "Fast direct super-resolution by simple functions," in *ICCV*, 2013, pp. 561–568. 20

[187] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, "Coupled dictionary training for image super-resolution," *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3467–3478, 2012. 20

[188] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *CVPR*. IEEE, 2008, pp. 1–8. 20

[189] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010. 20, 54, 58, 59, 63, 85

[190] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *ICLR*, 2016. 69

[191] K. Zeng, J. Yu, R. Wang, C. Li, and D. Tao, "Coupled deep autoencoder for single image super-resolution," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–11, 2016. 21

[192] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International conference on curves and surfaces*. Springer, 2010, pp. 711–730. 70

[193] ——, "On single image scale-up using sparse-representations," in *Curves and Surfaces*, 2012, pp. 711–730. 57

[194] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, 2017. 17, 52, 53, 58

[195] K. Zhang, X. Gao, D. Tao, and X. Li, "Multi-scale dictionary for single image super-resolution," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1114–1121. 20

[196] L. Zhang, W. Dong, D. Zhang, and G. Shi, "Two-stage image denoising by principal component analysis with local pixel grouping," *Pattern Recognition*, vol. 43, no. 4, pp. 1531–1549, 2010. 16

[197] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2226–2238, 2006. 18

[198] X. Zhu, X. Li, and S. Zhang, "Block-row sparse multiview multilabel learning for image classification," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 450–461, 2016. 33

[199] X. Zhu, L. Zhang, and Z. Huang, "A sparse embedding and least variance encoding approach to hashing," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 3737–3750, 2014. 33

[200] S. Zimmer, S. Didas, and J. Weickert, "A rotationally invariant block matching strategy improving image denoising with non-local means," in *International Workshop on Local and Non-Local Approximation in Image Processing*, 2008, pp. 135–142. 15

[201] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proceedings of IEEE International Conference on Computer Vision*, 2011, pp. 479–486. 2, 9, 15, 17, 24, 52, 53, 58, 118, 119, 120, 121, 122, 123, 124, 125, 126