

M.Sc. ENGG. THESIS

# Reduction of Mixed Gaussian-Impulse Noise Using Deep Convolutional Neural Network

by

Mohammad Tariqul Islam

Submitted to

Department of Electrical and Electronic Engineering

in partial fulfillment of the requirements for the degree of  
Master of Science in Electrical and Electronic Engineering



Department of Electrical and Electronic Engineering

Bangladesh University of Engineering and Technology (BUET)

Dhaka 1205

July 2018

# Declaration of Authorship

I, Mohammad Tariqul Islam, declare that this thesis titled, “Reduction of Mixed Gaussian-Impulse Noise Using Deep Convolutional Neural Network” or any part of it has not been submitted elsewhere for the award of any degree or diploma and that all sources are acknowledged.







---

Mohammad Tariqul Islam  
Candidate

The thesis titled “Reduction of Mixed Gaussian-Impulse Noise Using Deep Convolutional Neural Network”, submitted by Mohammad Tariqul Islam, Student-ID: **0416062209 P**, Session: April 2016, has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Science in Electrical and Electronic Engineering on July 18, 2018.

## Board of Examiners

1.   
Dr. S. M. Mahbubur Rahman  
Professor  
Department of Electrical and Electronic Engineering  
Bangladesh University of Engineering and Technology  
Dhaka-1205  
Chairman  
(Supervisor)
2.   
Dr. Md. Shafiqul Islam  
Professor and Head  
Department of Electrical and Electronic Engineering  
Bangladesh University of Engineering and Technology  
Dhaka-1205  
Member  
(Ex-Officio)
3.   
Dr. Md. Kamrul Hasan  
Professor  
Department of Electrical and Electronic Engineering  
Bangladesh University of Engineering and Technology  
Dhaka-1205  
Member
4.   
Dr. M. Sohel Rahman  
Professor  
Department of Computer Science and Engineering  
Bangladesh University of Engineering and Technology  
Dhaka-1205  
Member

# Acknowledgement

I would like to convey my heart-felt gratitude and thanks to my thesis supervisor Dr. S. M. Mahbubur Rahman, Professor of the Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Dhaka for the continuous guidance and support I have received from him during the course of this work. He was there with his kind help and direction anytime I was in need of them. His inspiration, motivation and friendly attitude helped make my work under him a great experience. It was a great pleasure to be a part of his research group and work under him.

I would also like to express my sincere appreciation to the members of my thesis committee, Dr. Md. Shafiqul Islam, Dr. Md. Kamrul Hasan and Dr. M. Sohel Rahman, for their valuable feedback and suggestion on my work.

In addition, my heart-felt thank goes to all the faculty members and my course teachers for their continuous help and guidance throughout the course of my M.Sc. program. I also thank my family members, without whose support and encouragement, it would be impossible to carry on with my study smoothly. I also acknowledge the help I got from Shuvajit Das, then President, BUET Photographic Society, for helping me for providing the images that I had used during the initial phase of the research. I also than Rafat Jamal Tazim for his help with the Technocampus hand NIR images database.

We also acknowledge the support of NVIDIA Corporation for the donation of a Titan Xp GPU that was used in this research.

*Dedicated to my loving parents*

# Abstract

The removal of mixed-noise is an ill-posed problem due to high level of non-linearity in the distribution of noise. Most commonly encountered mixed-noise is the combination of additive white Gaussian noise (AWGN) and impulse noise (IN) that have contrasting characteristics. A number of methods from the cascade of IN and AWGN reduction to the state-of-the-art sparse representation have been reported to reduce this common form of mixed-noise. In this thesis, a new learning-based algorithm using the convolutional neural network (CNN) models are proposed to reduce the mixed Gaussian-impulse noise from images. The models are evaluated for both the image to image learning as well as image to residual learning techniques. The proposed CNN models adopts computationally efficient transfer learning approach to obtain an end-to-end map from noisy image to noise-free image. The model has a small structure yet it is capable of providing performance superior to that of the well established methods. Experimental results on different settings of mixed-noise show that the proposed CNN image to image learning based denoising method performs significantly better than the sparse representation and patch-based methods do both in terms of accuracy and robustness. Moreover, due to the lightweight structure, the denoising operation of the proposed CNN-based method is computationally faster than that of the previously reported methods. The proposed image to residual learning based densely connected denoising CNN (DCDCNN) outperforms the previous state-of-the-art CNN based denoising method. Qualitative evaluation shows that the proposed DCDCNN produces visually superior images than the traditional as well as other CNN based methods. Despite being a deeper neural network architecture, the proposed DCDCNN can denoise in a very short time by employing GPU.

# Contents

<b>Abstract</b>	<b>v</b>
<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>xiv</b>
<b>List of Abbreviations</b>	<b>xvii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Related Works . . . . .	2
1.3 Scope of Analysis . . . . .	5
1.4 Specific Contributions . . . . .	6
1.5 Problem Formulation . . . . .	8
1.6 Outline . . . . .	9
<b>2 CNN: A Brief Review</b>	<b>10</b>
2.1 Introduction . . . . .	10
2.2 Layers of CNN . . . . .	10
2.2.1 Convolution Layer . . . . .	10
2.2.2 Batch Normalization Layer . . . . .	11
2.2.3 Rectified Linear Unit Layer . . . . .	12
2.2.4 Max-Pool Layer . . . . .	13

2.2.5	Dense Block . . . . .	13
2.3	Training . . . . .	14
2.3.1	Loss Functions for Classification . . . . .	15
2.3.2	Loss Functions for Regression . . . . .	16
2.3.3	Optimization Algorithms . . . . .	18
<b>3</b>	<b>Denoising Using Image to Image Learning</b>	<b>21</b>
3.1	Introdcution . . . . .	21
3.2	Image to Image Learning Model . . . . .	21
3.3	Training Scheme . . . . .	26
<b>4</b>	<b>Denoising Using Image to Residual Learning</b>	<b>28</b>
4.1	Introdcution . . . . .	28
4.2	Image to Residual Learning Model . . . . .	29
4.3	Training Scheme . . . . .	32
<b>5</b>	<b>Experiments and Results</b>	<b>35</b>
5.1	Introdcution . . . . .	35
5.2	Datasets . . . . .	36
5.3	Data Augmentation . . . . .	38
5.4	Experiments on Image to Image Denoising . . . . .	39
5.4.1	Model Setup . . . . .	39
5.4.2	Learned Filters . . . . .	42
5.4.3	Methods Used for Comparison . . . . .	44
5.4.4	Results . . . . .	47
5.5	Experiments on Image to Residual Denoising . . . . .	56
5.5.1	Model Setup . . . . .	56
5.5.2	Learned Filters . . . . .	59
5.5.3	Methods used for Comparison . . . . .	60
5.5.4	Results . . . . .	62



<b>6 Conclusions</b>	<b>72</b>
6.1 Future Works . . . . .	74
<b>Appendix</b>	<b>75</b>

# List of Figures

1.1	Effect of mixed AWGN-SPIN noise on a typical image and its histogram. (a) Noise-free image, (b) noisy image, (c) histogram of noise-free image, and (d) histogram of noisy image. . . . .	7
2.1	A dense block with three filter units. The inputs and outputs are colored in order to show the data flow. . . . .	14
3.1	Stick diagram of the proposed feed-forward CNN architecture showing the pre-processing steps and 4-stage convolution filtering. Operations from the left to right is considered to be the forward path. Each of the processing steps or layers of the network is represented by its corresponding geometric shape. The rank order filtering operation is shown by a stripe in a rectangle, the upsampling operation through bicubic interpolation by a diverging trapezoid, the convolution layer by a rectangle, the ReLU layer by a solid line, and the max-pooling layer by a converging trapezoid. . . . .	22
3.2	Denoising performance by varying number of stages in the CNN architecture. (a) Increase of PSNR from previous stage. (b) Increase of SSIM from previous stage. (c) Standard deviation of PSNRs. (d) Standard deviation of SSIMs. . . .	24
3.3	Block diagram of the proposed method showing rank order filter, upsampling employing bicubic interpolation and the CNN employing image to image learning method. . . . .	25

4.1	Proposed filter unit $H_{l_d}(\cdot)$ for denoising where the $1 \times 1$ convolution performs an expansion of channels and the $3 \times 3$ convolution layers perform compression of channels in the network. The convolution layers are referred to as CONV and the batch normalization layers are referred to as BN. . . . .	29
4.2	Proposed DCDCNN by employing the convolution, ReLU, batch normalization layers and dense block in a residual learning strategy. The convolution layers are referred to as CONV and batch normalization layers are referred to as BN. The learned residual image is subtracted from the input noisy image to obtain the denoised image. . . . .	31
4.3	Comparison of denoising performance for 11 commonly referred test images in terms of PSNR of a traditional method Cai's [1]+BM3D [2], DnCNN [3] validated using images contaminated by AWGN-IN, DnCNN validated using images contaminated by AWGN, and proposed DCDCNN validated using images contaminated by AWGN and optimized by employing squared Frobenius norm, SSIM loss function and both squared Frobenius norm and SSIM loss function, respectively, for removal of mixed AWGN+SPIN with noise parameters $\sigma = 25$ and $p = 0.15$ . . . . .	33
5.1	Learning curves of the proposed CNN-based model trained for removal of mixed AWGN+SPIN with noise parameters $\sigma = 10$ and $p = 0.30$ . The pink dotted learning curve is obtained when both the ROF and BI layers as well as the subsequent $MP_{K_1}$ layer are removed from the model. The solid red curve with circle markers results in when the network uses the ROF layer without the BI layer. The blue solid curve is obtained when the network uses both the ROF and BI layers but without any prior information for initialization of weights and biases. The black dashed curve is obtained when both the ROF and BI layers are employed and at the same time transfer learning is adopted using the known weights and biases that are trained for $\sigma = 10$ and $p = 0.15$ . . . . .	36

5.2	Grid of the filters in the filter set $\mathbf{W}_1$ learned from the training dataset for SPIN parameters $\sigma = 10$ and $p = 0.30$ . The kernel size of each of the filters is $7 \times 7$ . The filters are sorted according to the variance of coefficients. . . . .	39
5.3	The input and outputs of different convolution layers for AWGN+SPIN parameters $\sigma = 10$ and $p = 0.30$ . (a) Input noisy image. Typical ten outputs of each column are obtained from the convolution layers of (b) first stage, (c) second stage, and (d) third stage. (e) Output of the final convolution layer. . . . .	40
5.4	Visual comparison of the denoising performance of the methods for the test image <i>Boat</i> . (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 10$ and $p = 0.30$ (PSNR: 10.65 dB, SSIM: 0.0731). (c) The image is obtained by using Cai's method (PSNR: 27.70 dB, SSIM: 0.6929). The estimated noise-free images are obtained by using the methods (d) Cai's+BM3D [2] (PSNR: 31.21 dB, SSIM: 0.8529), (e) WESNR [4] (PSNR: 29.56 dB, SSIM: 0.8062), and (f) proposed CNN (PSNR: 31.92 dB, SSIM: 0.8596). . . . .	49
5.5	Visual comparison of the denoising performance of the methods on a high quality image having a close-up view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 25$ and $p = 0.15$ (PSNR: 12.40 dB, SSIM: 0.1419). (c) The image is obtained by using Cai's method (PSNR: 20.40 dB, SSIM: 0.3587). The estimated noise-free images are obtained by using the methods (d) Cai's+BM3D [2] (PSNR: 28.71 dB, SSIM: 0.8838), (e) WESNR [4] (PSNR: 25.77 dB, SSIM: 0.8458), and (f) proposed CNN (PSNR: 28.81 dB, SSIM: 0.8906). . . . .	50
5.6	Visual comparison of denoising performance of the proposed method for the color version of the high quality image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 10$ and $p = 0.30$ (PSNR: 5.47 dB, SSIM: 0.2466). (c) The estimated noise-free image is obtained by using the proposed CNN (PSNR: 24.79 dB, SSIM: 0.9296). . . . .	51

5.7	Visual comparison of the denoising performance of the methods for mixed AWGN+SPIN+RVIN denoising for an image having long shot view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN+RVIN with parameters $\sigma = 10$ , $p = 0.20$ and $r = 0.05$ (PSNR: 12.75 dB, SSIM: 0.2316). The estimated noise-free images are obtained by using the methods (c) AMF+ACWMF+BM3D [2] (PSNR: 22.64 dB, SSIM: 0.6893), (d) WESNR [4] (PSNR: 22.44 dB, SSIM: 0.6592), and (e) proposed CNN (PSNR: 22.83 dB, SSIM: 0.7088). . . . .	52
5.8	Comparison of execution time of the four experimental methods required to denoise an image of size $128 \times 128$ contaminated by mixed AWGN+SPIN with parameters $\sigma = 10$ and $p = 0.30$ . . . . .	53
5.9	Learning curves of the proposed DCDCNN trained for removal of mixed AWGN+SPIN with noise parameters $\sigma = 25$ and $p = 0.15$ . The pink dotted learning curve is obtained employing the DnCNN architecture. The solid red curve with circle markers results in DCDCNN architecture is trained employing SSIM loss function. The blue dashed curve is obtained when the DCDCNN architecture is trained using Frobenius norm. The black solid curve is obtained when the DCDCNN is trained employing both the Frobenius norm and SSIM loss function.	55
5.10	Grid of the filters in the filter set of the first convolutions layers of the proposed DCDCNN learned from the training dataset for AWGN+SPIN parameters $\sigma = 25$ and $p = 0.15$ . The kernel size of each of the filters is $3 \times 3$ . The filters are sorted according to the variance of coefficients. . . . .	57
5.11	The input and outputs of different convolution layers for mixed AWGN+SPIN removal using DCDCNN with noise parameters $\sigma = 25$ and $p = 0.15$ . (a) Input noisy images. All 64 outputs of the (b) first and (c) second convolution layers. (d) Typical 64 outputs of the second dense block. (e) The 64 outputs of the final transition layer. (f) Output of the final convolution layer which estimates the residual image. (g) The estimated denoised image (PSNR: 28.76 dB, SSIM: 0.8505). (h) The ground truth image. (i) The ground truth residual image. . . .	58

5.12	(a) Gaussian fit of the distributions of the PSNRs for DnCNN [3] ( $\mu = 27.28$ dB and $\sigma = 1.82$ ) and DCDCNN ( $\mu = 27.56$ dB and $\sigma = 2.10$ ). (b) Gaussian fit of the distributions of the SSIMs for DnCNN [3] ( $\mu = 0.8353$ and $\sigma = 0.0496$ ) and DCDCNN ( $\mu = 0.8449$ and $\sigma = 0.0518$ ). . . . .	62
5.13	Visual comparison of the denoising performance under heavy noise of the methods for a high quality image having a close-up view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 40$ and $p = 0.15$ (PSNR: 11.72 dB, SSIM: 0.1226). The estimated noise-free images are obtained by using the methods (c) Cai's [1]+BM3D [2] (PSNR: 26.81 dB, SSIM: 0.8269), (d) IIL [5] (PSNR: 24.50 dB, SSIM: 0.7705), (e) DnCNN [3] (PSNR: 26.73 dB, SSIM: 0.8282), and (f) proposed DCDCNN (PSNR: 27.45 dB, SSIM: 0.8612). . . . .	67
5.14	Visual comparison of the denoising performance for the color version of the high quality image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 10$ and $p = 0.30$ (PSNR: 9.86 dB, SSIM: 0.2245). The estimated noise-free images are obtained by using the methods (d) IIL [5] (PSNR: 30.06 dB, SSIM: 0.9686), (e) DnCNN [3] (PSNR: 30.71 dB, SSIM: 0.9725), and (f) proposed DCDCNN (PSNR: 31.5015 dB, SSIM: 0.9757). . . . .	68
5.15	Visual comparison of the denoising performance of the methods for commonly referred <i>Parrot</i> image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters $\sigma = 25$ and $p = 0.15$ (PSNR: 12.30 dB, SSIM: 0.1201). The estimated noise-free images are obtained by using the methods (c) Cai's [1]+BM3D [2] (PSNR: 27.80 dB, SSIM: 0.8326), (d) IIL [5] (PSNR: 27.75 dB, SSIM: 0.8268), (e) DnCNN [3] (PSNR: 28.56 dB, SSIM: 0.8411), and (f) proposed DCDCNN (PSNR: 28.77 dB, SSIM: 0.8503). . .	69

5.16	Visual comparison of the denoising performance of the methods for a near infrared (NIR) image. (a) The original noisy image. The image is denoised assuming noise parameters $\sigma = 20$ and $p = 0.15$ by (b) Cai's [1]+BM3D [2], (c) WESNR [4], (d) IIL [5], (e) DnCNN [3], and (f) proposed DCDCNN. . . . .	70
5.17	Comparison of execution time of the four experimental methods required to denoise an image of size $128 \times 128$ contaminated by mixed AWGN+SPIN with parameters $\sigma = 25$ and $p = 0.15$ . . . . .	70

# List of Tables

5.1	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 1. . . . .	43
5.2	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 2. . . . .	44
5.3	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 3. . . . .	45
5.4	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN+RVIN from Test Sets 1, 2, and 3. . . . .	46
5.5	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN and AWGN+SPIN+RVIN from commonly-referred five test images. . . . .	47
5.6	Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing AWGN from Test Set 1. . . . .	48
5.7	Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 1. . . . .	61
5.8	Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 2. . . . .	63
5.9	Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 3. . . . .	64



5.10	Denoising performance in terms of mean of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN with $\sigma = 25$ and $p = 0.15$ from commonly-referred test images. . . . .	65
5.11	Denoising performance in terms of mean of PSNR (in dB) and SSIM for reducing AWGN from commonly referred test images. . . . .	66

# List of Abbreviations

ACWMF	Adaptive Center Weighted Median Filter
Adam	Adaptive Momentum
AMF	Adaptive Median Filter
AWGN	Additive White Gaussian Noise
BI	Bicubic Interpolation
BM3D	Block Matching and 3D Filtering
BN	Batch Normalization
CNN	Convolutional Neural Network
DCT	Discrete Cosine Transform
DET	Object Detection
DnCNN	Denoising Convolutional Neural Network
DCDCNN	Densely Connected Denoising Convolutional Neural Network
IIL	Image to Image Learning
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IN	Impulse Noise
IRL	Image to Residual Learning
MF	Median Filter
MLP	Multi Layer Perceptron
MP	Max Pool
NLM	Non Local Means
NN	Neural Network
PSNR	Peak Signal to Noise Ratio
ReLU	Rectified Linear Unit
RMSProp	Root Mean Square Propagation
ROF	Rank order Filter
RVIN	Random Valued Impulse Noise
SGD	Stochastic Gradient Descent
SPIN	Salt and Pepper Impulse Noise
SSIM	Structural Similarity
WESNR	Weighted Encoding with Sparse Nonlocal Regularization

# Chapter 1

## Introduction

### 1.1 Introduction

Image denoising is a fundamental problem in image processing and computer vision. Images are corrupted during image acquisition or transmission due to inherent characteristics of imaging devices and transmission paths as well as due to defective equipment [6]. The goal of image denoising is to estimate the original noise-free image from its noisy observation. Image denoising as well as closely related operations like image inpainting [7] and watermark removal [8] are also recognized as preprocessing tasks such as image segmentation and pattern recognition in computer vision. Two commonly encountered noise types in the literature are the additive white Gaussian noise (AWGN) and impulse noise (IN). The AWGN, which primarily originates from the sensor temperature and illumination levels of the environment, affects the entire set of pixels of an image [9]. On the other hand, the IN caused by faulty sensors or transmission errors, replaces certain image pixels with random values. The reasons for the generation of these noises are common and they often occur simultaneously resulting in mixed form of AWGN and IN. For example, AWGN and IN occur simultaneously in digital photography due to sensor temperature and faulty sensor triggering [10]. The noise in imaging of computed tomography is often modeled as mixed AWGN-IN (see, for example, [11]). The complex mixing of photon and electronic noise, laser light reflection and dust on the glass slides are common in cDNA microarray imaging. In such a case, the noise in microarray images can be efficiently modeled as

mixed AWGN-IN [12]. The noise in near-infrared (NIR) and hyperspectral images is modeled as mixed noise consisting of AWGN, shot noise and IN [13]. The denoising of images itself is regarded as an ill-posed inverse problem; and noticeably the estimation of a noise-free image becomes more challenging in the case of mixed-noise due to the fact that the distributions of the Gaussian and impulse noise differ significantly. Reduction of these noises have been studied extensively in the past decades separately as well as in the mixed form. In this section, the denoising methods for the AWGN, IN and mixed Gaussian-impulse noise are briefly reviewed. Finally, the scope in the area and the contributions of the work are given.

## 1.2 Related Works

AWGN is widely studied noise model in the area of image denoising. Samples of a zero-mean Gaussian distribution are assumed to be added to pixels resulting in image corrupted with AWGN. A very common form of assumption for such a type of noise is the independent and identically distributed (i.i.d.) nature of the additive components. A good number of approaches have been investigated to restore images contaminated with AWGN. The noise can be reduced by using a simple approach of Gaussian filtering, in which case, however, the edge is not preserved giving rise to edge displacement, edge vanishing and even phantom edges [14]. Adaptive Gaussian filtering, bilateral filtering (BF), and non-local means (NLM) [15] consider this issue and show incremental accuracy in edge preservation. Since the inception of NLM, this approach has been rigorously investigated in a number of methods (see for example, [2, 16]). Multiresolution analysis has also been used to reduce AWGN from images because of its success in 1D signal denoising [17, 18]. For example, Rahman *et al.* [19] used the modified Gaussian-Hermite distribution to statistically model the discrete wavelet coefficients of images to reduce AWGN through Bayesian framework. Image denoising methods by using wavelets with improved directional selectivity such as the directionlet [20] and contourlet [21] were also tried. Dabov *et al.* [2] introduced block matching and 3D-filtering (BM3D) algorithm in which the similar image patches are grouped into a 3D matrix. Then, such groups are processed in a 3D transformed domain using one of the available sparse representations such as the DCT, bi-

orthogonal and Haar wavelet coefficients. Liu *et al.* [16] prescribed a method using an adaptive soft-thresholding to find similar patches in images along with the  $l_2$ -norm-based estimation for noise-free sparse coefficients. It was recommended that the block matching of BM3D should be carried out along edge direction to improve the denoising performance [22]. Xu *et al.* [23] developed a fast method based on the nonlocally centralized sparse representation algorithm. The neural network (NN) has also been employed to remove AWGN from images (see for example, [24, 25, 26]). The reduction of AWGN using the convolutional neural network (CNN) was recommended in [24]. Burger *et al.* [25] have provided a comparative study between the approaches of multilayer perceptron (MLP) and BM3D. Wang and Morel [27] used the mean-shift of the noisy patches and then attempted an MLP trained on a fixed strength noise level to remove Gaussian noise of variable strengths. Recently, a residual learning strategy has been adopted in CNN to reduce AWGN from images corrupted with unknown noise level [3].

Impulse noise is considered to be the replacement of certain portion of total pixels of an image by a set of fixed-level intensities with a given probability. Widely encountered impulse noise are salt and pepper impulse noise (SPIN) and random valued impulse noise (RVIN). In the case of SPIN, the corrupted image pixels get set at extreme values of the dynamic range of the image pixels, whereas a corrupted pixel in RVIN may be any of the random values within the range. In order to restore the images corrupted by IN, a number of nonlinear techniques have been tried. A simple rank order filter (ROF), e.g., the median filter (MF) [9] can be applied to detect and eliminate IN, but the local structures of images are destroyed by employing such a simple filtering technique for heavy noise corruption. In order to mitigate such problems, various improvements in MF such as the weighted MF [28], and center weighted MF (CWMF) [29] have been proposed. In general, these filters operate only on the corrupted pixels of an image, thus requiring a method to detect the noisy and noise-free pixels. The denoising methods using such an approach include the switching MF [30], adaptive MF (AMF) [31], and adaptive CWMF (ACWMF) [32]. The MLP scheme has also been used to restore images corrupted by IN (see, for example, [25]).

Tackling the mixture of AWGN and IN is relatively difficult because of the unique nature of each of the two types of noise. In general, the additive filters are successful in reducing

the Gaussian noise, while the order statistics filters for the impulse noise [33]. In order to reduce the mixed Gaussian-impulse noise from images, conventional methods use rank order statistics of pixels to detect and remove IN first, and then use a separate denoising technique by assuming that the residual noise is Gaussian. The order statistics filters when applied to images corrupted with mixed AWGN-IN produce grainy and visually unpleasant results, and successive Gaussian filters cannot compensate such noise effectively [34]. The alpha-trim mean filters provide both the additive and rank order properties. However, under heavy noise, these methods provide inadequate performance as the local structures of images become smeared. In order to preserve the details in an image, the BF has been extended as a trilateral filter to reduce the mixed-noise by incorporating the rank-ordered statistics of the absolute differences of intensities of neighboring pixels [34]. Cai *et al.* [35, 1] proposed a two-phase method to estimate the noise-free images. In this method, IN is detected and removed first by using AMF, and then the resultant image is denoised by optimizing an  $l_1$ -norm-based regularization function. Cai's method can be treated as a rank order filtering technique, provided the regularization parameter is very small. This method has been studied for mixed-noise removal by exploring different kinds of norms and regularization techniques in the second phase. The norms that have been investigated in the optimization technique include the total variation norm of wavelet coefficients of images [36]. Liu *et al.* [37] used the dictionary learning model with sparse representation to estimate the noise-free image. Agostinelli *et al.* [26] employed the ensemble of stacked sparse autoencoders and adaptive averaging technique for image denoising. Zhang *et al.* [38] proposed an iterative split-Bregman-based denoising algorithm that requires joint statistical modeling of similar patches in an image. In a unified framework of joint detection of noisy pixels and reduction of noise components, Xiang *et al.* [4] proposed an iterative dictionary learning-based method called the weighted encoding with sparse nonlocal regularization (WESNR). The method provides a good performance in preserving the local structures for relatively smooth images, but falls short when images have higher details. Dictionary learning method has recently been used for hyperspectral image super resolution under mixed Poisson-Gaussian noise [39].

### 1.3 Scope of Analysis

To the best of our knowledge, the removal of mixed-noise from images has not yet been attempted by using CNN, which has shown immense success not only in image classification [40] but also in different regression-type problems such as image enhancement [41], super-resolution [42], blind deconvolution [43], and inpainting [44]. The increasing interest of CNN is also largely due to the availability of application-oriented large databases and efficient parallel computing in graphics processing units (GPUs) [40]. Convolutional neural network has been attempted to remove AWGN from images (see [24], and [3]). Two stream NNs have also been employed for reducing AWGN and IN independently by training the patches of certain test images [26]. The success of CNN in many image processing techniques as well as a few instances of application of NNs in simple problems of image denoising has thus motivated us to develop a single-stream CNN architecture in order to tackle the highly challenging problem of removal of mixed-noise by generating the reconstruction filters with the consideration of large-scale image variabilities. A question may arise as to why CNN should be chosen as compared to the acclaimed denoising methods such as those that adopt sparse representation and patch based denoising. To answer this question, first we would like to refer to the methods adopting the sparse representation. Recently, it has been shown that CNN possesses higher representation capability as compared to traditional sparse representation, thus provides better performance in image super-resolution [42]. Traditionally, the sparse dictionaries are constructed by vectorizing the image matrices or patches, and thus 2D structural information, i.e., dependency of pixels of local neighboring regions may be lost in dictionary learning. On the contrary, CNN is capable of maintaining the 2D structural information both in the training and testing phases, since the convolution operation considers the local neighboring image pixels by using 2D masks. Second, the patch-based methods such as NLM and BM3D use self-similarity regularization and thus require a computationally heavy iterative optimization algorithm. In addition, the performance of these methods can be sub-optimal, if the images have a low number of self-similar patches. On the contrary, CNN optimizes its weights of the convolution masks through gradient-based training scheme, which inherently considers self-similarity in the entire set of

patches available in relatively large number of training images. In the training phase of a CNN-based network, the weights of the masks are learned through the gradients of local neighboring pixels in such a way that the noise is reduced. Since a huge number of samples as well as large variabilities in images are used during training, the CNN-based denoising is expected to be optimal. Moreover, the performance of CNN has been shown to have significantly improved due to the inclusion of newer activation functions like the rectified linear unit (ReLU) [40] and newer learning algorithms such as the root mean square propagation (RMSProp) [45]. Such a success has not been observed in other types of neural networks. For example, the long short term memory network [46], which is primarily used to model time series data, has not benefited much due to ReLU activation function, as CNN has. In addition, regression type tasks including denoising require images to be processed in patches prior to inputting them to the network. In such a case, the implementation of CNN using modern GPUs to train an end-to-end model is more efficient than that of MLP. Further, once a CNN is trained, the weights learned by the network can be transferred to a closely related network that has a similar setting for faster learning [47]. In addition there is scope of developing new denoising algorithm employing the image to residual learning [48] technique. Thus, the removal of mixed-noise using CNN is a viable idea and the development of efficient method of noise reduction using a suitable network architecture is worth investigating.

## 1.4 Specific Contributions

The main objective of this thesis is to present CNN architectures for reduction of mixed-noise encountered in practice. Overall, the contributions of this paper are as follows:

- A four-stage CNN architecture employing image to image learning is proposed for reduction of mixed Gaussian-impulse noise in images. The network learns end-to-end mapping from noisy images to noise-free estimates with trivial pre-processing.
- A very deep densely connected denoising convolutional neural network (DCDCNN) is proposed employing image to residual learning technique for the purpose of reducing



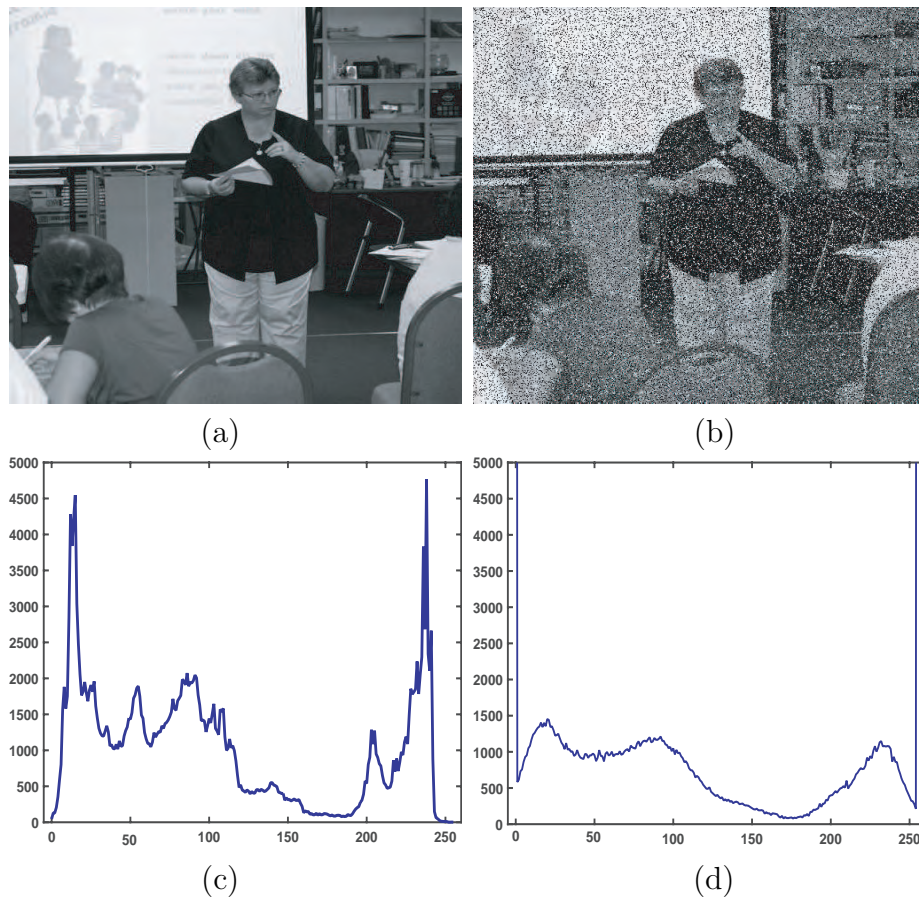


Figure 1.1: Effect of mixed AWGN-SPIN noise on a typical image and its histogram. (a) Noise-free image, (b) noisy image, (c) histogram of noise-free image, and (d) histogram of noisy image.

mixed Gaussian-impulse noise in images. The network learns end-to-end mapping from noisy images to residual noise without any pre-processing.

- Faster training of network is obtained by adopting the mechanism of transfer learning.
- The performance of the proposed CNN-based method is evaluated using sufficiently large datasets as well as commonly-referred test images. The overall denoising performance in terms of accuracy and robustness is shown to be better than that of the state-of-the-art methods considering different settings of mixed-noise.

## 1.5 Problem Formulation

Let  $\mathbf{X}$  be a noise-free image of size  $M_v \times M_h$  with each element being represented as  $x(m_v, m_h)$  at pixel location  $(m_v, m_h)$ , where  $m_v \in \{1, 2, \dots, M_v\}$  and  $m_h \in \{1, 2, \dots, M_h\}$ . Let  $\mathbf{X}_n$  be the noisy observation of the image  $\mathbf{X}$  with a relation given by

$$\mathbf{X}_n = f(\mathbf{X}) \quad (1.1)$$

where  $f(\cdot)$  is the degradation function. Also, let a noisy pixel be denoted as  $x_n(m_v, m_h)$ . We consider two types of corruption: 1) mixed AWGN and SPIN and 2) mixed AWGN, SPIN and RVIN. If the image is corrupted only by AWGN, then a noisy pixel is given by

$$x_n(m_v, m_h) = x(m_v, m_h) + \nu(m_v, m_h) \quad (1.2)$$

where  $\nu(m_v, m_h)$  is a sample of i.i.d. zero-mean Gaussian distribution with standard deviation  $\sigma$ . Within the given dynamic range, let the maximum and minimum values of an image pixel be  $d_{max}$  and  $d_{min}$ , respectively. Then, an image is corrupted with SPIN when  $x_n(m_v, m_h)$  is either  $d_{max}$  or  $d_{min}$  with equal probability  $p/2$  ( $p \leq 1$ ). Thus, for mixed AWGN+SPIN noise a pixel is corrupted by AWGN with a probability  $(1 - p)$ . Using these definitions, the noisy observation for each pixel of mixed AWGN+SPIN can be described as [4]

$$x_n(m_v, m_h) = \begin{cases} d_{min} & \text{with probability } p/2 \\ d_{max} & \text{with probability } p/2 \\ x(m_v, m_h) + \nu(m_v, m_h) & \text{with probability } 1 - p \end{cases} \quad (1.3)$$

In a similar fashion, let an image be corrupted with RVIN when  $x_n(m_v, m_h)$  attains a random value  $d(m_v, m_h)$  with probability  $r$  ( $r \leq 1$ ). The value  $d(m_v, m_h)$  is uniformly distributed within the dynamic range  $[d_{min}, d_{max}]$ . Using the definition of RVIN, the noisy observation for

each pixel of mixed AWGN+SPIN+RVIN can be described as [4]

$$x_n(m_v, m_h) = \begin{cases} d_{min} & \text{with probability } p/2 \\ d_{max} & \text{with probability } p/2 \\ d(m_v, m_h) & \text{with probability } r(1-p) \\ x(m_v, m_h) + \nu(m_v, m_h) & \text{with probability } (1-p)(1-r) \end{cases} . \quad (1.4)$$

Figure 1.1 shows a typical image and its noisy version corrupted with AWGN and SPIN, and the corresponding histograms of the images. The noisy image is obtained by using the AWGN parameter  $\sigma = 10$  and SPIN parameter  $p = 0.3$ . It is observed from this figure that the degradation function changes the original histogram significantly. In particular, the variation of histogram in the dynamic region has been smoothed significantly primarily due to AWGN. At the same time, two strong peaks have appeared at the two ends of the histogram due to the presence of SPIN. In other words, the effects of mixed-noise on the histogram of image are contrasting in nature. The goal of denoising is to find an estimate  $\hat{\mathbf{X}}$  for the noise-free image  $\mathbf{X}$  from its noisy observation  $\mathbf{X}_n$ .

## 1.6 Outline

The rest of the thesis is organized as follows. Chapter 2 provides a brief review of the CNN architecture and the training schemes employed in the literature. Chapter 3 provides a description of the proposed image to image learning (IIL) based denoising scheme. The proposed image to residual learning (IRL) based denoising scheme is discussed in Chapter 4. The datasets, the representations learned by the proposed networks and the experimental results along with comparison to other state-of-the-art methods have been discussed in Chapter 5. This chapter also shows the effectiveness of the proposed methods by varying the noise parameters and observing visual outputs. Finally, the conclusions are provided in Chapter 6 where the findings of the study are summarized.

# Chapter 2

## CNN: A Brief Review

### 2.1 Introduction

In machine learning convolutional neural network (CNN) [49] is a class of deep learning algorithm [50]. It is a type of feed-forward artificial neural network where the filtering is performed employing convolution kernels. Such a network can be employed for both classification and regression. In this chapter, a brief review of the CNN architecture and its training methods are described.

### 2.2 Layers of CNN

The typical layers of a CNN are the convolution layer performing the filtering operation, the ReLU layer performing the activation, and the max-pool layer reducing the spatial dimensions. Recently, with the introduction of batch normalization and dense connection, the performance of the networks have further increased. The layers are described in brief as follows:

#### 2.2.1 Convolution Layer

This layer performs the 2D convolution operation on the input data  $\mathbf{X}_{i-1}$  using a set of layer-dependent filters. Let  $\mathbf{W}_i$  be a filter set with dimension  $C_i \times C_{i-1} \times N_i \times N_i$ , where  $C_i$  and  $C_{i-1}$

are the number of channels of the output and input of this layer, respectively, and  $N_i$  is the square-size support parameter of the filters. The parameter  $C_i$  represents the number of filters in the set  $\mathbf{W}_i$ . Each of the filters has a corresponding bias term, resulting in a bias vector  $\mathbf{b}_i$  with  $C_i$  number of elements. Hence, the output of this layer is obtained from the output of the previous layer, the bias term, and the corresponding filter set as

$$\mathbf{X}_i = \mathbf{W}_i * \mathbf{X}_{i-1} + \mathbf{b}_i \quad (2.1)$$

where  $*$  represents the linear convolution operation. This operation results in the dimension of output  $\mathbf{X}_i$  to be  $C_i \times M_{vi} \times M_{hi}$  from input  $\mathbf{X}_{i-1}$  with shape  $C_{i-1} \times M_{v(i-1)} \times M_{h(i-1)}$ . There is a parameter called ‘stride’, which can be used in the convolution layer to increase or decrease the spatial dimensions of the output. The spatial dimensions remain the same from the input to the output, when the parameter is set to unity. A value of the stride parameter greater than unity decreases the dimensions of the output, whereas a value less than unity increases the dimensions. In the case of dimensionality reduction in the output, however, a general tendency is to set the stride parameter to unity and to use the pooling layer to perform the downsampling operation.

### 2.2.2 Batch Normalization Layer

Batch normalization (BN) layer performs normalization on the input data  $\mathbf{X}_{i-1}$  using the average  $\mu_B$  and standard deviation  $\sigma_B^2$  of the input data. The output of the layer is obtained by [51]

$$x_{m,i} = \gamma_s \frac{x_{m,i-1} - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta_s \quad (2.2)$$

where learnable parameters  $\gamma_s$  and  $\beta_s$  are called scale and shift parameter, respectively. During training, running average and running standard deviation are employed for  $\mu_B$  and  $\sigma_B^2$ ,

respectively and are given by [51]

$$\mu_B = \frac{1}{M_B} \sum_{m=1}^{M_B} x_{m,i-1} \quad (2.3)$$

$$\sigma_B^2 = \frac{1}{M_B} \sum_{m=1}^{M_B} (x_{m,i-1} - \mu_B)^2 \quad (2.4)$$

$$(2.5)$$

where  $M_B$  is the number of elements in the input. During testing the population mean and population standard deviation are employed. Without batch normalization a network requires lower learning rate and carefully tuned initialization. Moreover, using such normalization the outputs never saturates and always stays within small values and thus the network converges quickly.

### 2.2.3 Rectified Linear Unit Layer

It is referred to as the activation layer, wherein only the elements of input with non-negative values are transmitted to the output and the rest are set to zero. In other words, the input-output relation of this layer is given by

$$\mathbf{X}_i = \max(\mathbf{0}, \mathbf{X}_{i-1}) \quad (2.6)$$

where  $\mathbf{0}$  is a zero-matrix with same size as that of  $\mathbf{X}_i$  by considering the fact that the dimensions of the input and output of this layer remain the same. It is well known that the rectification aids the convergence of the neural network much faster as compared to the traditional sigmoid function does. In addition, the ReLU unit assists the neural network to attain a better sparse representation (see [52]). It is customary that the convolution layer or batch normalization layer be followed by the ReLU activation.

### 2.2.4 Max-Pool Layer

In this layer, a non-linear downsampling operation is carried out between the input and the output. A mask of size  $K_i \times K_i$  is selected along the spatial dimensions of the input in a non-overlapping manner, and then the maximum value of the data in the mask is passed on to the output layer. The pooling layers are reported to increase the robustness of CNN-based algorithms in the presence of noise and clutter, and to be particularly well suited for attaining sparsity in the data [53]. In practice, the max-pool layer is preceded by a ReLU layer. To describe this layer, the notation  $MP_{K_i}$  is used in the model description.

### 2.2.5 Dense Block

This block a combination of several layers where the outputs of each of the past convolution layers in the block are concatenated [54, 55]. The output of the  $l_d^{th}$  convolution layer in a dense block is given by

$$\mathbf{X}_{l_d} = H_{l_d}([\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_3, \dots, \mathbf{X}_{l_d-1}]) \quad (2.7)$$

where  $l_d^{th} \in 1, 2, \dots, L_D$  and  $L_D$  is the total number of filter units in the dense block,  $\mathbf{X}_0$  is the input to the dense block,  $[\mathbf{X}_0, \mathbf{X}_1, \mathbf{X}_3, \dots, \mathbf{X}_{l_d-1}]$  is the concatenation of the past outputs, also called, dense connection, and the function  $H_{l_d}(\cdot)$  is the functional representation of the filter unit of the dense block. In the original implementation  $H_{l_d}(\cdot)$  employed batch normalization layer followed by ReLU activation and convolution layer [55]. Sometimes, each of the  $H_{l_d}(\cdot)$  functions are preceded by a bottleneck layer, which is a function similar to  $H_{l_d}(\cdot)$  but employs  $1 \times 1$  convolution layer with lower number of filters. Bottleneck layers are used to improve computational efficiency by reducing number of channels in the output [56]. Figure 2.1 shows a diagram of a dense block with three filter units, i.e.,  $L_D = 3$ . It is seen from the figure that, the output of the dense block contains the input as well as the intermediate outputs of the filter units. It is to be noted that, if each of the  $H_{l_d}(\cdot)$  units have a receptive size of  $N_R$ , then a dense block with three filter units provides an output with contains the data processed with

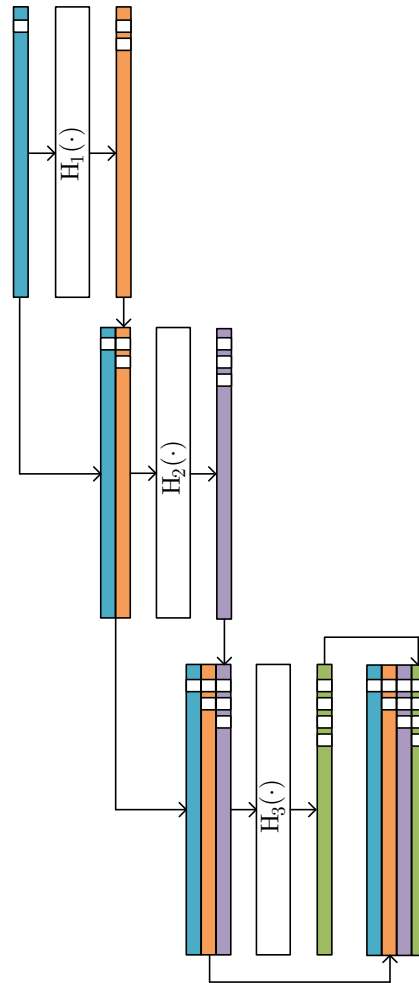


Figure 2.1: A dense block with three filter units. The inputs and outputs are colored in order to show the data flow.

receptive size of  $0$ ,  $N_R$ ,  $2 \times (N_R - 1) + 1$  and  $3 \times (N_R - 1) + 1$ . Thus, it is expected that the future layers are able to better understand the dataflow and can process the data with optimal receptive size.

## 2.3 Training

In order to map from input to output the trainable weights and parameters have to be computed using a training process. The most common method to train a CNN is to minimize a loss function employing the backpropagation algorithm. The loss function indicates how far the



state of the network is from the desired outcome. The backpropagation algorithm provides a direction to which the trainable parameters have to be changed in order to minimize the value of the loss provided by the loss function. The information obtained using backpropagation algorithm is employed in an optimization algorithm to update the state of the network. In this section, the cost functions for classification and regression analysis, and the minimization algorithms are briefly described. It is to be noted that, the loss discussed here are for single training sample, which is extended for a mini-batch such that, the mini-batch loss is the average of loss of each of the samples.

### 2.3.1 Loss Functions for Classification

For a  $(\mathbf{x}_i, y_i)$  pair of input and label, in classification task, the NN estimates the label of the input from a pre-determined set of output labels  $(1, 2, \dots, L_N)$ , where  $L_N$  is the total number of labels in the set. The output of an NN provides scores  $s_1, s_2, \dots, s_{L_N}$  against each of the labels. Usually the label that holds the highest score is considered to be the estimated label by the network. The loss function in a classification analysis, provides a measure as to how this scores are evaluated. In this section, the multiclass Support Vector Machine (SVM) and the cross-entropy loss functions for classification analysis are discussed.

#### SVM Loss Function

The SVM loss function defines the loss as a sum of distance between the correct class and incorrect classes, subject to a threshold and is given by [57]

$$D_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta) \quad (2.8)$$

where  $s_j$  is the score of the  $j^{\text{th}}$  class,  $s_{y_i}$  is the score of the correct class provided by the neural network and  $\Delta$  is a hyperparameter that provides the expected distance between the two scores. It is seen from (2.8) that the loss function is a sum which is computed only on the incorrect classes. It is expected that the sum  $s_j - s_{y_i} + \Delta \leq 0$  which only happens when the score of the

correct class is greater than the score of the incorrect classes by at least  $\Delta$ . This formulation is also known as hinge loss or Weston Watkins formulation.

### Cross-entropy Loss Function

The cross-entropy loss function considers the scores as the unnormalized log probabilities and defined as,

$$D_i = -\log \left( \frac{e^{s_{y_i}}}{\sum_j e^{s_j}} \right) \quad (2.9)$$

where the function  $f(x) = \frac{e^{x_i}}{\sum_j e^{x_j}}$  is called the softmax function. Thus, this formulation is also known as softmax classifier. The softmax function can be interpreted as the normalized probability assigned to the correct class. Thus the negative sign in (2.9) indicates that when loss function is minimized, the probability of the correct class is maximized. The final target is to obtain probability of 1 for the correct label, and 0 for the other labels, thus in turn to obtain a Kroneker delta function.

### 2.3.2 Loss Functions for Regression

For a pair of input and desired output  $(\mathbf{x}_i, y_i)$ , in regression task, the output of the NN  $\hat{y}_i$  estimates the desired output  $y_i$  as closely as possible. Thus, unlike classification, where the output scores of the network ranks the labels and the label has to be chosen from those scores, in regression, the output of network is the expected output of the network. Thus, common distance metrics such as L1, L2 can be employed as loss functions, and a specialized loss function image processing using structural similarity (SSIM) [58] image quality index can be employed to define a distance from the input to the output. In this section, these functions are described briefly.

### L1 Loss Function

L1 loss function, also known as the least absolute deviation, is given by

$$D_i = |y_i - \hat{y}_i|. \quad (2.10)$$

It is seen from (2.10) that the value of the loss is always positive and provides a measure of distance from the desired output and estimated output. The objective is to minimize the value of the loss to 0.

### L2 Loss Function

L2 loss function, also known as the least square deviation, is given by

$$D_i = \|y_i - \hat{y}_i\|^2 = (y_i - \hat{y}_i)^2. \quad (2.11)$$

Similar to L1, the L2 loss provide measure of distance between the desired and estimated output. However, L2 loss penalizes large distance with larger loss compared to L1 loss. Thus, it is often opined that, for image processing L1 loss provides a smoother output compared to L2 loss. It is noted that, L2 loss has a inverse relation to image quality metric peak signal to noise ratio (PSNR).

In the case where the output of a neural network is a matrix or a tensor  $\hat{\mathbf{Y}}_i$  instead of scalar as in (2.11), the loss is referred to as squared Frobenius loss function and for a desired output  $\mathbf{Y}_i$  is given by

$$D_i = \left\| \mathbf{Y}_i - \hat{\mathbf{Y}}_i \right\|_F^2 = \sum_j \sum_k |y_{i,j,k} - \hat{y}_{i,j,k}|^2 \quad (2.12)$$

where  $\|\cdot\|_F$  is the Frobenius norm. The minimum value of this loss function is zero.

### SSIM Loss Function

The SSIM loss function is defined strictly for matrix and tensors. SSIM is a method for estimating the perceived quality of an image. For a desired output  $\mathbf{Y}_i$  and estimated output  $\widehat{\mathbf{Y}}_i$ , the SSIM metric is given by

$$f_{\text{SSIM}}(\mathbf{Y}_i, \widehat{\mathbf{Y}}_i) = \frac{(2\mu_{\mathbf{Y}_i}\mu_{\widehat{\mathbf{Y}}_i} + c_1)(2\sigma_{\mathbf{Y}_i\widehat{\mathbf{Y}}_i}^2 + c_2)}{(\mu_{\mathbf{Y}_i}^2 + \mu_{\widehat{\mathbf{Y}}_i}^2 + c_1)(\sigma_{\mathbf{Y}_i}^2 + \sigma_{\widehat{\mathbf{Y}}_i}^2 + c_2)} \quad (2.13)$$

where  $\mu_{\mathbf{Y}_i}$  is the mean of  $\mathbf{Y}_i$ ,  $\mu_{\widehat{\mathbf{Y}}_i}$  is the mean of  $\widehat{\mathbf{Y}}_i$ ,  $\sigma_{\mathbf{Y}_i}^2$  is the variance of  $\mathbf{Y}_i$ ,  $\sigma_{\widehat{\mathbf{Y}}_i}^2$  is the variance of  $\widehat{\mathbf{Y}}_i$ ,  $\sigma_{\mathbf{Y}_i\widehat{\mathbf{Y}}_i}$  is the covariance of  $\mathbf{Y}_i$  and  $\widehat{\mathbf{Y}}_i$ , and  $c_1$  and  $c_2$  are constants to stabilize the division operation. Usually,  $c_1$  is set to  $(k_1L)^2$  and  $c_2$  is set to  $(k_2L)^2$ , where  $L$  is the dynamic range of the image pixels and  $k_1$  and  $k_2$  are constants and set to 0.01 and 0.03 in the original implementation [58].

As the SSIM index provides similarity between the desired and estimated output, the loss function is defined in terms of dissimilarity. The value of SSIM is between  $-1$  and  $+1$ . Thus the SSIM loss function in terms of dissimilarity is given by

$$D_i = \mathcal{L}_{\text{SSIM}}(\mathbf{Y}_i, \widehat{\mathbf{Y}}_i) = 1 - f_{\text{SSIM}}(\mathbf{Y}_i, \widehat{\mathbf{Y}}_i) \quad (2.14)$$

where  $\mathcal{L}_{\text{SSIM}}(\cdot)$  represents the SSIM loss function. When, the desired output  $\mathbf{Y}_i$  and estimated output  $\widehat{\mathbf{Y}}_i$  are identical, (2.14) attains the minimum value of 0.

### 2.3.3 Optimization Algorithms

A trainable parameter  $w$  is updated using the gradient of the loss function  $D$  with respect to  $w$ . A common approach is to iteratively update the value of the trainable parameter  $w(\eta)$  using the information of the gradient  $\frac{dD(\eta)}{dw(\eta)}$ , where  $\eta$  is the iteration number. The most common algorithms are stochastic gradient descent (SGD), momentum, root mean square propagation (RMSProp) and adaptive momentum (Adam) algorithm. The optimization algorithms are described briefly in this section.

### Stochastic Gradient Descent (SGD)

In its simplest form the SGD algorithm is given by

$$w(\eta + 1) = w(\eta) - \lambda \frac{dD(\eta)}{dw(\eta)} \quad (2.15)$$

where  $\lambda$  ( $\lambda > 0$ ) is a hyperparameter called learning rate.

### Momentum

It is an extension to SGD algorithm, there the equation tracks the momentum  $\Delta w$  in each iteration. The update is then conducted based on the gradient as well as  $\Delta w$  and is given by

$$\Delta w(\eta + 1) = \alpha \Delta w(\eta) - \lambda \frac{dD(\eta)}{dw(\eta)} \quad (2.16)$$

$$w(\eta + 1) = w(\eta) + \Delta w(\eta + 1) \quad (2.17)$$

where  $\lambda$  is learning rate and  $\alpha$  ( $0 < \alpha < 1$ ) is another hyperparameter called momentum. In most cases this approach provides a better convergence. The momentum part of (2.17) ensures that the algorithm updates the parameters in a direction which is between the direction of the previous step and the direction from the gradient step. As a result, for an erratic mini-batch the parameter stays within the expected value instead of off shooting due to erroneous gradient.

### Root Mean Square Propagation (RMSProp)

In this algorithm, the update of trainable parameter  $w$  is performed in terms of the weighted average of the square of the gradient, which is also referred to as the mean squared given by [45]

$$MS(w(\eta)) = \gamma MS(w(\eta - 1)) + (1 - \gamma) \left( \frac{\partial D(\eta)}{\partial w(\eta)} \right)^2 \quad (2.18)$$

$$w(\eta + 1) = w(\eta) - \frac{\lambda}{\sqrt{MS(w(\eta)) + \epsilon}} \frac{\partial D(\eta)}{\partial w(\eta)} \quad (2.19)$$

where  $\lambda$  and  $\gamma$  ( $0 < \gamma < 1$ ) are hyper-parameters known as the learning and decay rates, respectively, and  $\epsilon$  is a numerical stability factor. RMSProp is an attempt to modulate the learning rate for each of the parameters independently employing the mean square value.

### Adaptive Momentum (Adam)

In this algorithm, the update of the trainable parameter  $w$  is performed in terms of both the first ( $m$ ) and second momentum ( $v$ ) of the gradient  $\frac{dD(\eta)}{dw(\eta)}$ . In fact, the second moment is same as the mean square of RMSProp algorithm. Thus, Adam, an extension of the RMSProp algorithm, is given by [59]

$$m(\eta + 1) = \beta_1 m + (1 - \beta_1) \frac{dD(\eta)}{dw(\eta)} \quad (2.20)$$

$$v(\eta + 1) = \beta_2 v + (1 - \beta_2) \left( \frac{\partial D(\eta)}{\partial w(\eta)} \right)^2 \quad (2.21)$$

$$w(\eta + 1) = w(\eta) - \lambda \frac{m(\eta + 1)}{\sqrt{v(\eta + 1) + \epsilon}} \quad (2.22)$$

where  $\beta_1$  ( $0 < \beta_1 < 1$ ) and  $\beta_2$  ( $0 < \beta_2 < 1$ ) are hyperparameters called forgetting factor of first and second momentum of the gradient, respectively and other parameters have usual meaning. Adam algorithm in effect is a combination of both momentum update algorithm and RMSProp algorithm. The direction the parameter updates is conducted is defined by first momentum, which is, as previously, a direction between the direction of previous update and the direction of the gradient step. At the same time, the learning rate is modulated due to the second momentum or the mean squared value.

# Chapter 3

## Denoising Using Image to Image Learning

### 3.1 Introduction

Image to image learning (IIL) is a common technique for image enhancement and denoising [24, 25, 26, 42]. It has been employed for AWGN reduction [25, 26], natural image deformities removal [24] and image super-resolution [42]. Most of the methods employ NN in a direct image to image translation manner, whereas [42] employs pre-processing in order to improve the super resolution improvement. For reduction of mixed Gaussian-impulse noise from images using CNN image to image learning we thus employ application specific pre-processing which facilitates the denoising of mixed AWGN-IN [5]. This chapter provides a detail description and rationale of the pre-processing, CNN model and the training scheme employed for denoising mixed AWGN-IN from images using image to image translation.

### 3.2 Image to Image Learning Model

In general, a CNN-based method produces an end-to-end model with trivial pre- and post-processings, which are usually referred to as the non-trainable operations. In many cases, an application-oriented pre-processing can boost the performance of the neural network signifi-

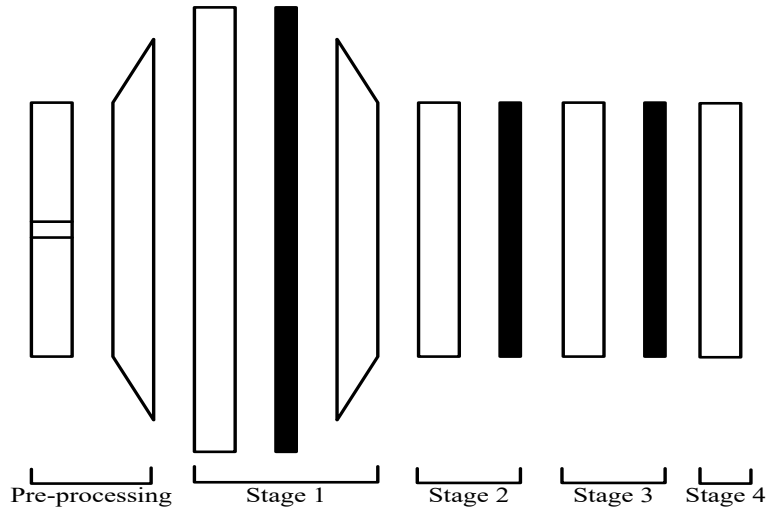


Figure 3.1: Stick diagram of the proposed feed-forward CNN architecture showing the pre-processing steps and 4-stage convolution filtering. Operations from the left to right is considered to be the forward path. Each of the processing steps or layers of the network is represented by its corresponding geometric shape. The rank order filtering operation is shown by a stripe in a rectangle, the upsampling operation through bicubic interpolation by a diverging trapezoid, the convolution layer by a rectangle, the ReLU layer by a solid line, and the max-pooling layer by a converging trapezoid.

cantly (see for example, the usage of bicubic interpolation applied for super-resolution in [42]). In the literature, almost all the methods use an ROF such as MF as a preprocessor to denoise images corrupted by mixed Gaussian-impulse noise. We have also adopted such a practice, and hence, the first step of the proposed model to denoise the corrupted image  $\mathbf{X}_0$  is given by

$$\mathbf{X}_1 = ROF(\mathbf{X}_0) \quad (3.1)$$

where  $ROF$  denotes a suitable rank order filter or combination of such filters. The ROF can be formed by using existing filters such as MF, AMF, CWMF, ACWMF or Cai's method [35] that are successful for reducing IN. The choice of such a filter largely depends on the type of IN corrupting the image. In [4], it is suggested that ROF be AMF when the noise is SPIN, and MF when the noise is SPIN+RVIN. In the proposed model, the Cai's method [35] has been chosen as ROF when the impulse noise is SPIN, and AMF followed by ACWMF has been chosen when the noise is SPIN+RVIN. In the stick diagram, we recommend the use of a stripe



in a rectangle to represent ROF.

To feed the convolution layer with a slightly smoother version of the noisy image, an upsampling operation is performed on the rank order filtered image using the bicubic interpolation (BI). In other words, the input-output relation of the second layer of the proposed model is given by

$$\mathbf{X}_2 = BI(\mathbf{X}_1) \quad (3.2)$$

where  $BI$  denotes the interpolation function. It can be shown that the frequency response of the interpolation functions exhibit a nature of a low-pass filter [60]. Thus, some high frequency components that arise from the rank order filtering on the Gaussian noise are mitigated using such functions. In the stick diagram, we prefer to use a diverging trapezoid to represent this interpolation function.

In order to denoise the rank order filtered and interpolated image, a 4-stage convolution filtering scheme is employed in the proposed model. A natural question that may arise is as to why 4 stages of convolutional layer are chosen in the model. In this context, an ablation study using 1000 images is performed by varying the number of convolutional layers from 2 to 6 in the proposed architecture. It is observed that the relative improvements of denoising performance in terms of mean of the peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) [58] decrease with the number of stages. Figure 3.2 shows the relative improvements of PSNR and SSIM in percentage as well as the standard deviation of these metrics as the number of stages increases. The dotted line in the figure shows the increasing or decreasing trend of the metrics. It is seen from this figure that the increase in PSNR with the addition of a new convolution layer is not so significant when the number of layers is more than 4. In particular, the increase in PSNR, i.e.,  $\Delta\text{PSNR}$ , due to the inclusion of a new convolutional layer over 4 is less than 0.5%. Similar observation is made for the increase in SSIM, i.e.,  $\Delta\text{SSIM}$ , due to the addition of a new convolution layer. Further, it is observed that increasing the number of convolutional layers not only increases the computational load significantly, but also decreases the robustness in terms of increasing the standard deviation of both the metrics.

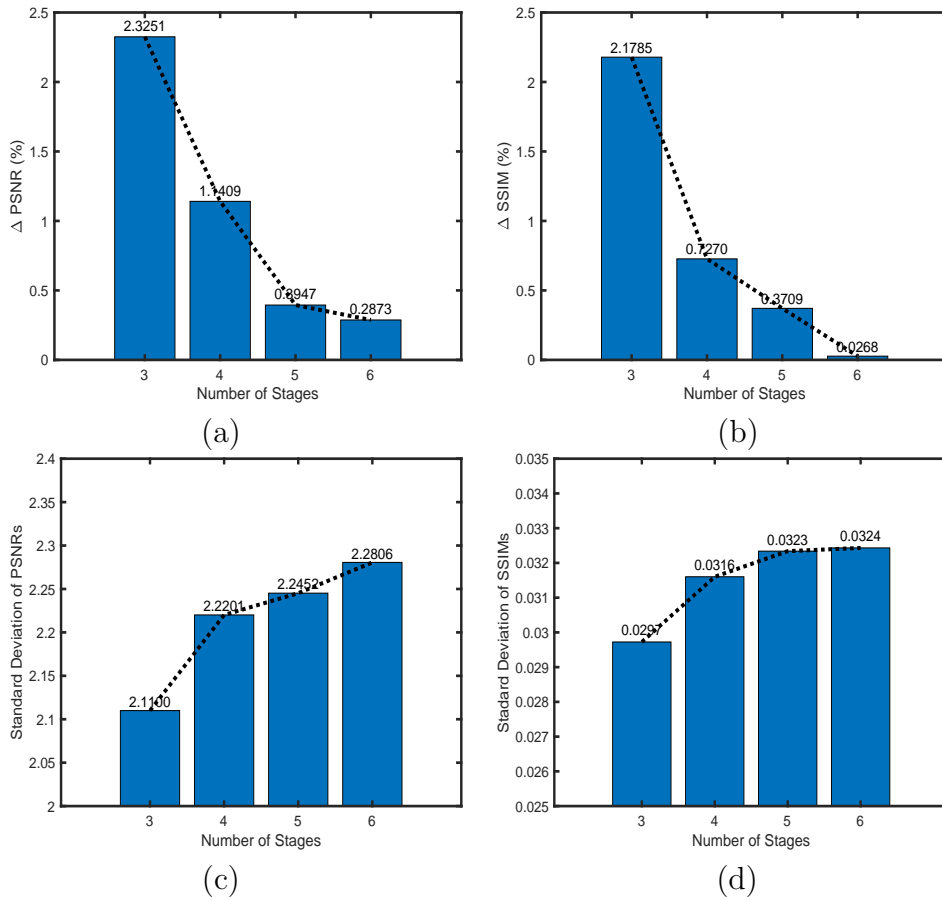


Figure 3.2: Denoising performance by varying number of stages in the CNN architecture. (a) Increase of PSNR from previous stage. (b) Increase of SSIM from previous stage. (c) Standard deviation of PSNRs. (d) Standard deviation of SSIMs.

Thus, it is concluded that a CNN architecture with 4 convolution layers is adequate to provide a satisfactory level of denoising performance both in terms of accuracy and robustness. In each of the four stages, the convolution layers may be followed by a ReLU or max-pool layer. Since the spatial dimensions of a noisy image are increased due to interpolation in the preprocessing step, at least one downsampling operation using the max-pool layer is required in the overall model. In this context, only the first stage convolution filtering uses all three kinds of layers. In particular, the output of the convolution layer is fed to a ReLU activation layer. This output is then max-pooled and spatial dimension is reduced. Thus, the output of the first stage CNN

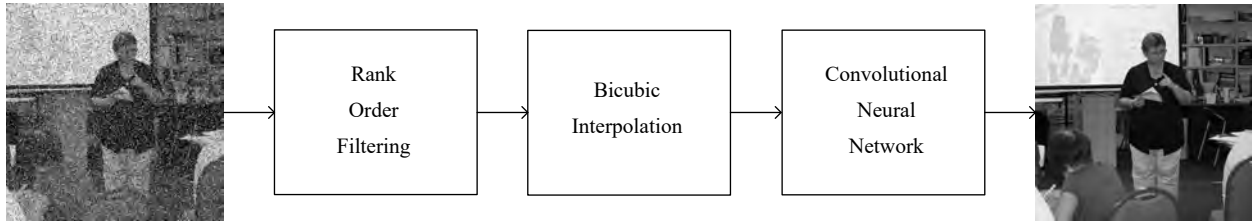


Figure 3.3: Block diagram of the proposed method showing rank order filter, upsampling employing bicubic interpolation and the CNN employing image to image learning method.

filtering can be written as

$$\mathbf{X}_3 = MP_{K_1}(\max(\mathbf{0}, \mathbf{W}_1 * \mathbf{X}_2 + \mathbf{b}_1)). \quad (3.3)$$

It is evident that the spatial dimensions of the output of the first stage convolution filtering are the same as that of the noisy input image. The second stage of the proposed model uses only the convolution and ReLU layers providing the output given by

$$\mathbf{X}_4 = \max(\mathbf{0}, \mathbf{W}_2 * \mathbf{X}_3 + \mathbf{b}_2). \quad (3.4)$$

The third stage of the network is a repetition of the second, and thus, the output is given by

$$\mathbf{X}_5 = \max(\mathbf{0}, \mathbf{W}_3 * \mathbf{X}_4 + \mathbf{b}_3). \quad (3.5)$$

In the final stage, only the convolution layer is used in the model. This stage provides the estimate of the noise-free image having the same dimensions as that of input as

$$\hat{\mathbf{X}} = \mathbf{W}_4 * \mathbf{X}_5 + \mathbf{b}_4. \quad (3.6)$$

Figure 3.1 shows the stick diagram of the proposed feed-forward CNN-based image denoising method with distinct marking of the preprocessing steps and 4-stage convolutional filtering. The geometric shapes of the preprocessing steps have been specified in the beginning of this section, whereas that of the CNN layers follow the convention. It is seen from this figure that the architecture of the proposed CNN model is relatively small, and thus, it is implementable even

in a light weight computational system. Figure 3.3 shows the block diagram of the proposed method. The ROF, BI pre-processing operations are shown separately. The CNN is employed in image to image learning scheme where the loss function is set to minimize the loss of the estimated image which is described in the training scheme.

### 3.3 Training Scheme

In order to obtain the end-to-end mapping function to denoise the images, the parameters of the proposed CNN model are required to be evaluated. The filters  $\mathbf{W}_1$ ,  $\mathbf{W}_2$ ,  $\mathbf{W}_3$  and  $\mathbf{W}_4$  and bias terms  $\mathbf{b}_1$ ,  $\mathbf{b}_2$ ,  $\mathbf{b}_3$  and  $\mathbf{b}_4$  in the four convolution layers of the proposed model are the parameters that have to be determined through learning. In order to train the network with the back-propagation algorithm, a differentiable and tractable loss function is required. The choices of loss function for such a regression task include the  $l_1$  norm,  $l_2$  norm, and Frobenius norm. The Frobenius norm is defined for matrices, and thus suits well when working with images. Moreover, the square of the norm is preferable due to numerical stability of the differentiation. Thus, we choose the squared Frobenius norm as the loss function in order to learn the network, and therefore, to estimate the parameters of the mapping function.

In general, in order to learn the CNN model, different types of data augmentation techniques are adopted during the training session [40]. This is mainly due to the fact that an optimal set of parameters of the network are achieved by increasing the number of image variabilities seen by the network. The augmented set of noisy images are fed to CNN and the corresponding noise-free images are estimated. Then, the loss function is defined in terms of an original image  $\mathbf{X}$  and the corresponding estimate  $\hat{\mathbf{X}}$  as

$$D = \left\| \mathbf{X} - \hat{\mathbf{X}} \right\|_F^2 \quad (3.7)$$

where  $\| \cdot \|_F$  is the Frobenius norm. Elements of the convolution filters, denoted by  $w$ , are updated using the gradient of the loss function  $D$  with respect to  $w$ . The update is performed in terms of the weighted average of the square of the gradient, which is also referred to as the

mean squared given by [45]

$$MS(w(\eta)) = \gamma MS(w(\eta - 1)) + (1 - \gamma) \left( \frac{\partial D(\eta)}{\partial w(\eta)} \right)^2 \quad (3.8)$$

$$w(\eta + 1) = w(\eta) - \frac{\lambda}{\sqrt{MS(w(\eta)) + \epsilon}} \frac{\partial D(\eta)}{\partial w(\eta)} \quad (3.9)$$

where  $\lambda$  ( $\lambda > 0$ ) and  $\gamma$  ( $0 < \gamma < 1$ ) are hyper-parameters known as the learning and decay rates, respectively,  $\eta$  is the iteration number, and  $\epsilon$  is a numerical stability factor. In a similar fashion, the elements of the bias terms are also updated in the training phase. It is noted that once a CNN is trained for a certain set of noise parameters, the filters and bias terms learned by the network can be used to initialize the same with new set of noise parameters, and thus achieving faster learning of the networks [47].

# Chapter 4

## Denoising Using Image to Residual Learning

### 4.1 Introduction

Image enhancement using image to residual learning is a relatively recent innovation [61] and have been explored in few applications such as image super-resolution [61] and AWGN removal [3]. Kim *et al.* [61] employed the idea of residual learning [48] for image to image translation. In this method, the output of a NN is constructed employing a residual connection which forces the network to learn the difference between the input image and expected output image, i.e. residual image. Zhang *et al.* [3] employed a similar idea by introducing directly learning the residual image instead of a residual connection and used batch normalization in intermediate layers for AWGN denoising. However, there is still scope of improvement in the data flow of the network, network architecture and training methods. In this chapter, we propose a CNN based image to residual learning method and improve the architectural design in terms of data flow and training method for reducing mixed AWGN-IN from design.

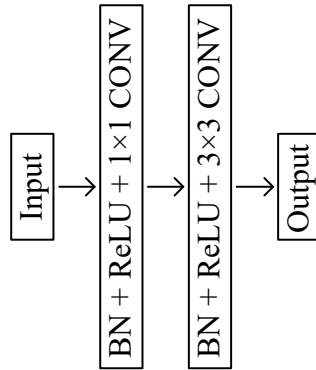


Figure 4.1: Proposed filter unit  $H_{l_d}(\cdot)$  for denoising where the  $1 \times 1$  convolution performs an expansion of channels and the  $3 \times 3$  convolution layers performs compression of channels in the network. The convolution layers are referred to as CONV and the batch normalization layers are referred to as BN.

## 4.2 Image to Residual Learning Model

In general, the target of a deep learning based method is to map the input and output using as little pre-processing as possible. In this context, Denoising CNN (DnCNN) [3] provided a state-of-the-art performance in AWGN and AWGN-like noise removal. However, removing mixed AWGN-IN is relatively difficult task and there is scope of improvement in terms of architecture design and training method. Thus, we have taken DnCNN as a reference and incrementally added improvements in the framework pertinent to remove mixed AWGN-IN from images.

First, a DnCNN architecture is trained for removing mixed AWGN-IN with  $\sigma = 25$  and  $p = 0.15$ . The network is validated on a set of 6 images contaminated with same noise parameters as the training noise parameters and is optimized by employing squared Frobenius Norm as loss function and Adam optimizer and by learning the residual image. The model is validated after every 100 iterations and the model that has performed best on the validation set is chosen for testing purpose. The test images in this experiments contained 10 images. However, it is found that, the over all denoising performance is not optimal compared to simple traditional methods (refer to Figure 4.3). It is suspected that, either the DnCNN is not able to remove the ill posed mixed AWGN-IN or some techniques have to be employed in order to improve the denoising performance of this framework.

In the second attempt, instead of the validation set employing the same noise parameters

as the training noise parameters, a validation set of the same 6 images with noise parameters  $\sigma = 25$  and  $p = 0.00$  is employed, i.e., the validation set is contaminated with only AWGN noise. Thus instead of tracking the performance on mixed AWGN-IN removal, during validation the performance of the training is tracked only for AWGN removal. Upon testing the best validated network on the same test set that is used in the previous attempt, it is found that the denoising performance of mixed AWGN-IN improves significantly. Thus in the rest of the paper, the models are validated on a set of images contaminated with corresponding AWGN parameter of the training noise parameter.

The DnCNN employ a fixed receptive size, referred to as patch size in the corresponding paper, of  $35 \times 35$  in the architecture. A natural question arise, whether this receptive size is optimal or whether a variable receptive size can be employed so that the CNN can automatically select the optimal receptive size for denoising purpose. In this context, dense blocks discussed in Section 2.2.5 can be employed to obtain a variable receptive size for image processing. However, dense connections in a dense block increase in size upon successive layers. Thus, it is often recommended that, bottleneck layers are to be employed before the dense connection in order to reduce the number of inputs in subsequent dense connections [55]. In the original implementation, a large number of filters were employed for  $3 \times 3$  convolution in the filter units and low number of filters were employed for  $1 \times 1$  convolution in the bottleneck layers [55] in an expander and compressor settings. In a denoising task, the spatial dimension is relatively high and thus using higher number of filters in  $3 \times 3$  convolution would require a larger memory. Thus, in the proposed filter unit  $H_{I_d}(\cdot)$  of the dense block for denoising, we employ larger number of filters  $C_E$  for  $1 \times 1$  convolution layer referred to as expander and lower number of filters  $C_C$  for the  $3 \times 3$  convolution layers referred to as compressor, i.e.,  $C_E > C_C$ . Since, the two convolutional filter sets are  $1 \times 1$  and  $3 \times 3$  respectively, each filter unit has an effective receptive size of  $3 \times 3$ . Each of the convolution layers in the filter units are preceded by a batch normalization and a ReLU layer. Figure 4.1 shows the proposed filter unit showing the expander and the compressor convolution filters. One more benefit that of this arrangement ( $C_E > C_C$ ) is that the number of parameters and memory requirement in subsequent filter layers increase less explosively compared to the traditional arrangement ( $C_E < C_C$ ) employed



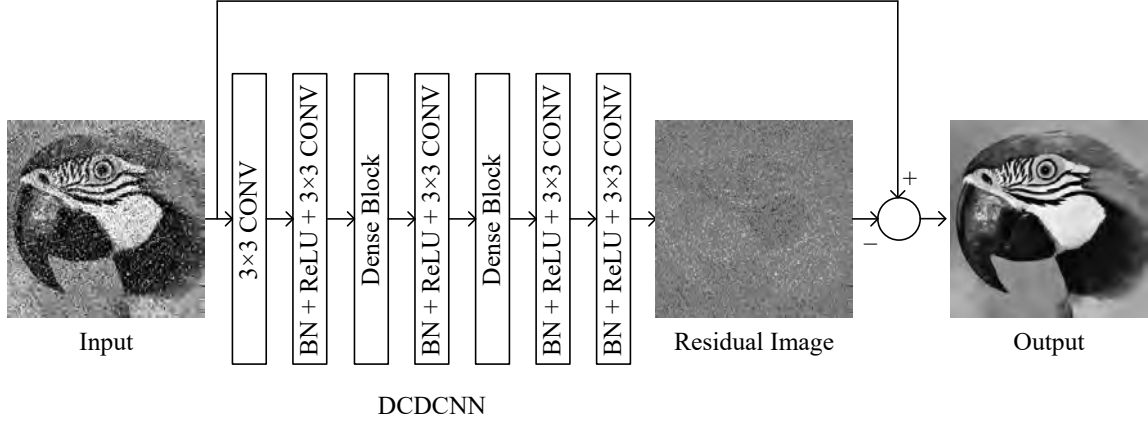


Figure 4.2: Proposed DCDCNN by employing the convolution, ReLU, batch normalization layers and dense block in a residual learning strategy. The convolution layers are referred to as CONV and batch normalization layers are referred to as BN. The learned residual image is subtracted from the input noisy image to obtain the denoised image.

for classification task.

In order to denoise mixed AWGN-IN employed the proposed dense block, a maximum filter size of  $3 \times 3$  is adopted as this provides a small incremental receptive size which are computationally efficient to train and are known to provide smooth outputs [62]. Similar to DnCNN, in the proposed architecture the spatial size of the input is never altered during any part of the architecture. Since data size, i.e., number of channels, inside a dense block increase in subsequent layers, instead of using one long dense block, two short dense blocks each with  $L_D$  filter units have been employed in the proposed CNN. Only the final output layer are not followed by any activation or batch normalization layer. Figure 4.2 shows the model architecture using a block diagram, which we refer to as Densely Connected Denoising Convolutional Neural Network or DCDCNN for short. DCDCNN employs the residual learning strategy which estimates the residual image  $\mathbf{R}$  from the input corrupted image  $\mathbf{X}_0$  given by

$$\hat{\mathbf{R}} = f_{\text{DCDCNN}}(\mathbf{X}_0) \quad (4.1)$$

where  $f_{\text{DCDCNN}}(\cdot)$  represents the mapping function given by the proposed DCDCNN model, and  $\hat{\mathbf{R}}$  is the estimated residual image. The estimate of the noise free image  $\hat{\mathbf{X}}$  is then given

by

$$\hat{\mathbf{X}} = \mathbf{X}_0 - \hat{\mathbf{R}} \quad (4.2)$$

It is found that, instead of using residual connection to impose residual learning as in [61], directly learning the residual image using (4.1) and then estimating the noise free image using (4.2) provided a better denoising performance.

### 4.3 Training Scheme

In order to learn the end-to-end mapping function of input noisy image to the estimated residual image by the proposed DCDCNN, the optimal values of the learnable parameters, i.e., the weights and biases of the convolutional filters and scales and shift parameters of the batch normalization filters, have to be evaluated. In order to train the network with the back-propagation algorithm, a differentiable and tractable loss function is required. The choices of loss function for such a regression task include the  $l_1$  norm,  $l_2$  norm, Frobenius norm and the SSIM loss function. The Frobenius norm is defined for matrices, and thus suits well when working with images. Moreover, the square of the norm is preferable due to numerical stability of the differentiation. Thus, we choose the squared Frobenius norm as the loss function in order to learn the network, and therefore, to estimate the parameters of the mapping function. However, it is shown that for improving perceptual quality of the images the perceptual loss functions such as SSIM would be helpful. In order to obtain image to image translation perceptual loss functions has been employed in the literature [63]. It is expected that such a loss function would improve the estimate of the residual image in image to residual learning as well. Thus we choose the SSIM loss function as a perceptual loss function in the proposed method. We define the overall loss function as a linear sum of the squared Frobenius norm and the SSIM loss function.

In general, in order to learn the CNN model, different types of data augmentation techniques are adopted during the training session [40]. This is mainly due to the fact that an optimal set

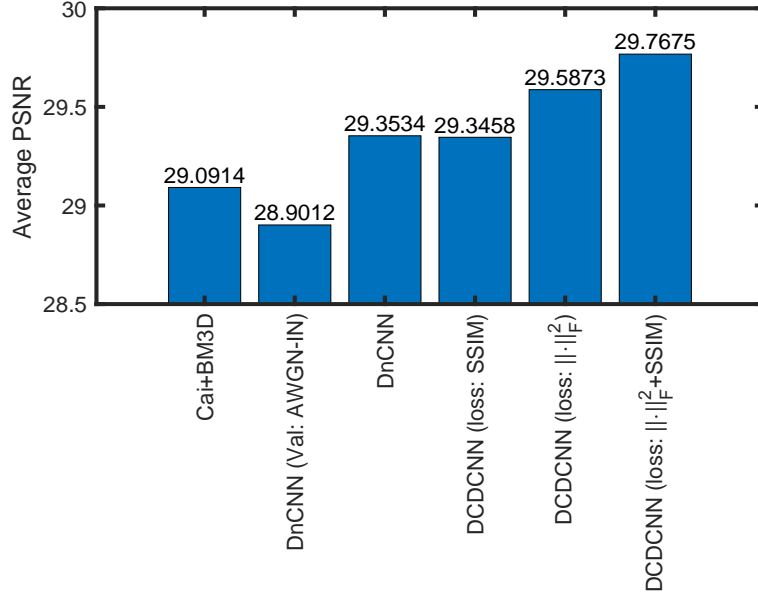


Figure 4.3: Comparison of denoising performance for 11 commonly referred test images in terms of PSNR of a traditional method Cai’s [1]+BM3D [2], DnCNN [3] validated using images contaminated by AWGN-IN, DnCNN validated using images contaminated by AWGN, and proposed DCDCNN validated using images contaminated by AWGN and optimized by employing squared Frobenius norm, SSIM loss function and both squared Frobenius norm and SSIM loss function, respectively, for removal of mixed AWGN+SPIN with noise parameters  $\sigma = 25$  and  $p = 0.15$ .

of parameters of the network are achieved by increasing the number of image variabilities seen by the network. The augmented set of noisy images are fed to CNN and the corresponding noise-free images are estimated. Then, the loss function is defined in terms of an original residual image  $\mathbf{R}$  and the corresponding estimate  $\hat{\mathbf{R}}$  as

$$D = \left\| \mathbf{R} - \hat{\mathbf{R}} \right\|_F^2 + \mathcal{L}_{\text{SSIM}}(\mathbf{R}, \hat{\mathbf{R}}) \quad (4.3)$$

where  $\|\cdot\|_F$  is the Frobenius norm and  $\mathcal{L}_{\text{SSIM}}(\cdot)$  is the SSIM loss function. Elements of the convolution filters, denoted by  $w$ , are updated using the gradient of the loss function  $D$  with respect to  $w$ . The update is performed in terms of both the first moment  $m$  and second momentum  $v$  of the gradient  $\frac{dD(\eta)}{dw(\eta)}$ , referred to as the Adam optimization algorithm given

by [59]

$$m(\eta + 1) = \beta_1 m + (1 - \beta_1) \frac{dD(\eta)}{dw(\eta)} \quad (4.4)$$

$$v(\eta + 1) = \beta_2 v + (1 - \beta_2) \left( \frac{\partial D(\eta)}{\partial w(\eta)} \right)^2 \quad (4.5)$$

$$w(\eta + 1) = w(\eta) - \lambda \frac{m(\eta + 1)}{\sqrt{v(\eta + 1) + \epsilon}} \quad (4.6)$$

where  $\beta_1$  ( $0 < \beta_1 < 1$ ) and  $\beta_2$  ( $0 < \beta_2 < 1$ ) are hyper-parameters called forgetting factor of first and second momentum of the gradient, respectively,  $\lambda$  ( $\lambda > 0$ ) is a hyper-parameter known as the learning rate,  $\eta$  is the iteration number, and  $\epsilon$  is a numerical stability factor. In a similar fashion, the elements of the bias terms are also updated in the training phase.

Figure 4.3 shows the denoising performance in terms of PSNR of a traditional method (Cai's [1]+BM3D [2]), DnCNN and proposed DCDCNN in different settings for removal of mixed AWGN+SPIN with noise parameters  $\sigma = 25$  and  $p = 0.15$  on a set of 11 common test images. As discussed previously DnCNN is employed in two settings, i.e., the validation set is contaminated with AWGN-IN, and the validation set is contaminated with only AWGN. In all of the DCDCNN experiments, the validation set is contaminated with AWGN alone and differs in terms of loss functions. The DCDCNN networks employ squared Frobenius norm, SSIM loss function and linear sum of both squared Frobenius norm and SSIM loss function. It is seen that, validating on mixed AWGN-IN has resulted in a poor performance for DnCNN which resulted in a denoising performance lower than the traditional method, whereas validating on images contaminated by AWGN only has resulted in a better mixed AWGN-IN denoising performance. It is also seen that, the DCDCNN performs best when sum of both squared Frobenius norm and SSIM loss function has been employed as the loss function.

# Chapter 5

## Experiments and Results

### 5.1 Introduction

Experiments are carried out to evaluate the performance of the proposed CNN-based methods by comparing it with that of the existing ones in reducing the mixed Gaussian-impulse noise from images. In this section, first we describe the datasets used in the experiments. Next, the data augmentation processes, which are required to achieve generalized results, are described. After that the results for image to image CNN-based method are described. The setup of the proposed CNN model and the initialization of the parameters are presented in the subsection. The representations of filters learned in the training phase are given in the following subsection. Then, the denoising methods that are compared with the proposed CNN-based method are described briefly. Next, the comparative results on the datasets in terms of the performance metrics, visual quality, and computational load are presented. Finally, the results for image to residual learning based DCDCNN are discussed. The setup of the proposed DCDCNN and the initialization of the parameters are presented. The filter characteristics and the representations learned by the trained filters are discussed in the following subsection. The setup of the proposed DCDCNN model and the initialization of the parameters are presented. Then, the comparing methods are described. And finally, the comparative results on the dataset in terms of the quantitative metrics, visual quality, and computational load are presented.

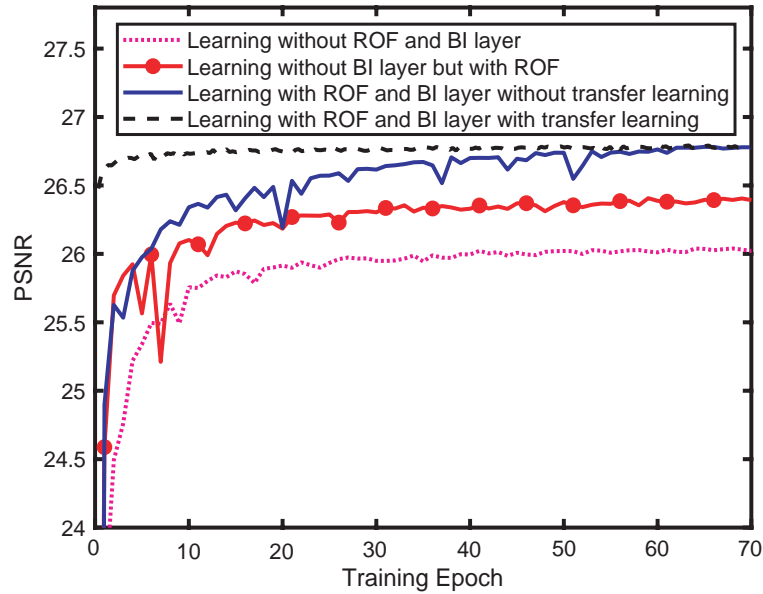


Figure 5.1: Learning curves of the proposed CNN-based model trained for removal of mixed AWGN+SPIN with noise parameters  $\sigma = 10$  and  $p = 0.30$ . The pink dotted learning curve is obtained when both the ROF and BI layers as well as the subsequent  $MP_{K_1}$  layer are removed from the model. The solid red curve with circle markers results in when the network uses the ROF layer without the BI layer. The blue solid curve is obtained when the network uses both the ROF and BI layers but without any prior information for initialization of weights and biases. The black dashed curve is obtained when both the ROF and BI layers are employed and at the same time transfer learning is adopted using the known weights and biases that are trained for  $\sigma = 10$  and  $p = 0.15$ .

## 5.2 Datasets

The construction of training and testing sets is very important for learning-based methods. A natural question arises as how to define these sets. In this context, first we would like to emphasize that the reporting of average denoising performance on a large set of test images is preferable over that of ordinary performance on a small number of recurrently appearing images in the literature. In other words, the testing sets should be relatively large so that one can have more realistic evaluations of the performance. Second, the deep learning produces generalized results when the training dataset is comprehensive. Thus, we ensure that the training dataset is sufficiently large. Finally, for a learning-based method, the performance should be more authentic, if the training and testing datasets are mutually exclusive and at the same time they are collected from different sources. In order to meet these requirements, we ensure

that both the training and testing datasets in the experiments are sufficiently large and they are constructed from two different databases. In particular, we consider the Places2 [64] and ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2015 object detection (DET) [65] databases, which are well known in the area of computer vision and image processing. The first database is chosen for training and validation, and the second one for the testing purpose. The experiments are conducted by converting the color images to grayscale images.

The training dataset is constructed from the Places2 database by randomly choosing  $2 \times 10^4$  images. It has been reported that a CNN model can reasonably learn from a smaller sample size ( $\sim 100$ ) for a regression-type problem as compared to a larger sample size ( $\sim 10^5$ ) for a classification-type problem (see, for example, [42]). Hence, the size of training set as considered in the experiment of image denoising is reasonable. In addition, to obtain generalized results, we adopt data augmentation techniques (see details in Section 5.3), which eventually increases the number of images several times in the training phase. In order to track the improvement of denoising after each epoch of the training phase, 50 mutually exclusive images are selected from the Places2 database as a validation set. Further, to facilitate the computation of the batch gradient descent technique used in CNN, it is required that the size of the images of the training and validation sets is the same, and preferably their sizes are small. In this context, each image from the training and validation sets is changed to a square shape by cropping it from the center while keeping the smaller dimension of the original image intact, and then the images are resized to  $128 \times 128$ . In the experiments on mixed AWGN-SPIN, the noisy versions of the images are generated by contaminating the processed images following (1.3). We have considered four values of the AWGN parameter, namely,  $\sigma = 10, 15, 20,$  and  $25$ , and two values of the SPIN parameter, namely,  $p = 0.15$  and  $0.30$  that result in eight different training and testing scenarios. For experiments involving the mixed AWGN+SPIN+RVIN, the noisy images are generated by using (1.4), wherein the noise parameters are chosen as  $\sigma = 10, p = 0.20$  and  $r = 0.05$ , and  $\sigma = 20, p = 0.15$  and  $r = 0.10$ .

In order to test the performance of the trained network, generic test datasets are constructed by choosing 1000 images from the ILSVRC 2015 DET database. The chosen images have good mixtures of smooth regions, edges, and textures. The test sets are obtained by considering

three practical scenarios described below:

1. Test Set 1: Similar to the training set, 1000 images are converted to grayscale images of size  $128 \times 128$ . The noisy test images are generated with noise parameters that are the same as for the training phase.
2. Test Set 2: The same images as in Test Set 1 are selected in this scenario, but the attributes of the noise parameters are chosen differently. This set analyzes the performance of the proposed method when the noise parameters in the testing phase deviate from that in the training phase. In particular, the parameters  $\sigma$ ,  $p$  and  $r$  are varied up to  $\pm 10\%$  from their nominal values randomly for each of the images. The values for  $\sigma$ ,  $p$ , and  $r$  are drawn from uniform distributions of  $\mathcal{U}(0.9\sigma, 1.1\sigma)$ ,  $\mathcal{U}(0.9p, 1.1p)$ , and  $\mathcal{U}(0.9r, 1.1r)$ , respectively, for each image.
3. Test Set 3: This test set is generated with a view to obtaining the performance of the network for arbitrary size images. In this scenario, 15 images are chosen from the testing set such that the dimension without resizing has the maximum variation with respect to the training set. In the experiments, the dimension of these images varies from 146 to 500 and 226 to 500 along the horizontal and vertical axes, respectively. The value of the noise parameters, namely,  $\sigma$ ,  $p$ , and  $r$ , are kept the same as that in the training dataset.

In order to investigate the denoising performance of the proposed method on individual images, the commonly-referred test images are also considered in the experiments.

### 5.3 Data Augmentation

Data augmentation results in the trained filters for denoising more robust to unseen images. In the experiments, a number of augmentation techniques are employed on the training images. The operations include the transpose, and the horizontal and vertical flips of the image matrices. The probability of each of the operations is set to 0.5. Thus, the combination of the operations produces up to 8 variants of each of the images. As a result, different orientations of the same



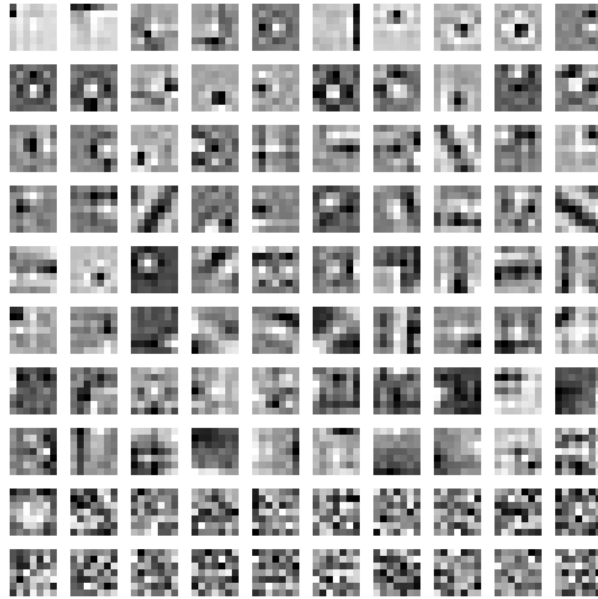


Figure 5.2: Grid of the filters in the filter set  $\mathbf{W}_1$  learned from the training dataset for SPIN parameters  $\sigma = 10$  and  $p = 0.30$ . The kernel size of each of the filters is  $7 \times 7$ . The filters are sorted according to the variance of coefficients.

image is available to the network for learning. Moreover, the samples of the noise sequence are altered after every 20 epochs in the training phase. Consequently, different instances of the noise for each of the training images are seen by the network. In effect, the size of the ultimate training dataset turns several times larger than that of initial set, which helps the network to attain generalization.

## 5.4 Experiments on Image to Image Denoising

### 5.4.1 Model Setup

The values of the parameters of the proposed CNN model described in Section 3.2 are chosen as per the dimensions of the input and output as well as those recommended in practice. Since the experimental images in the training and test datasets are grayscale in nature, the parameter  $C_0$  in the input channel at the very beginning is set to 1. The proposed four-stage network has four filter sets, namely,  $\mathbf{W}_1$ ,  $\mathbf{W}_2$ ,  $\mathbf{W}_3$  and  $\mathbf{W}_4$ , for which the number of filters in each set and that of the weights in each filter are assigned as per the number of output channels and the

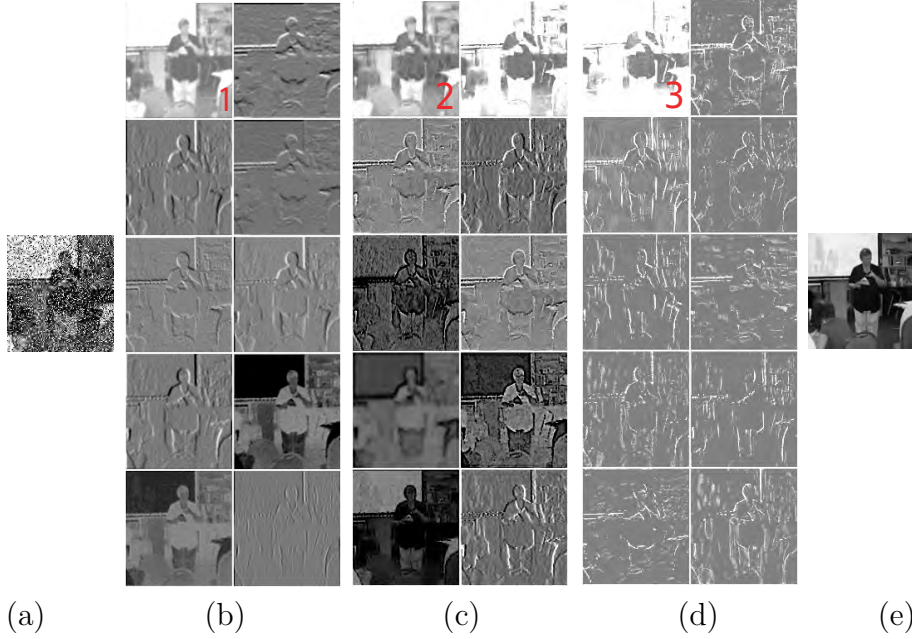


Figure 5.3: The input and outputs of different convolution layers for AWGN+SPIN parameters  $\sigma = 10$  and  $p = 0.30$ . (a) Input noisy image. Typical ten outputs of each column are obtained from the convolution layers of (b) first stage, (c) second stage, and (d) third stage. (e) Output of the final convolution layer.

kernel size used, respectively, in a layer. By the definition of a convolution layer, the number of output channels and that of the bias terms for the layer are kept equal. In the experiments, the channel parameters  $C_1$ ,  $C_2$  and  $C_3$ , corresponding to the filter sets  $\mathbf{W}_1$ ,  $\mathbf{W}_2$  and  $\mathbf{W}_3$ , are set to 100, 200 and 100 when  $\sigma < 20$  and to 100, 500 and 100 when  $\sigma \geq 20$ , respectively. The filter set  $\mathbf{W}_4$  is in the last stage of the network, and the corresponding channel parameter  $C_4$  specifies the output channel. This channel parameter is set to 1 as the output provides the estimated noise-free image of the corresponding noisy input. The kernel sizes of the filters are chosen based on the strength of Gaussian noise. It has been observed that a larger kernel size is beneficial in denoising for a higher value of  $\sigma$ . In addition, the kernel size  $N_1$  of the filter set  $\mathbf{W}_1$  is set to a higher value so that information can be extracted from a relatively larger neighboring region of an upscaled image in the first stage. The kernel size of the following three filter sets, namely,  $\mathbf{W}_2$ ,  $\mathbf{W}_3$  and  $\mathbf{W}_4$ , is set to a lower value as compared to  $N_1$  in order to process a smaller local region of the filtered input, which has the spatial dimensions the same as that of the original image. Independent of the noise strength, the quadruplet  $(N_1, N_2,$

$N_3, N_4$ ) is chosen as (7, 5, 5, 3). The parameter  $K_1$  of the max-pool layer is set to 2, which is a traditional value. This value of the parameter also results in the spatial dimensions of the final output to be the same as the original image from the upscaled version of the output of the first convolution and ReLU layers.

The initialization of the weights and biases of the network and that of the parameters of learning mechanism are important for efficient training. When no prior information about the weights  $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$  and  $\mathbf{W}_4$  and the corresponding biases  $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$  and  $\mathbf{b}_4$  is available, they are initialized by taking samples from a zero-mean Gaussian distribution with standard deviation  $10^{-3}$ . In order to slow down the gradient descent and to avoid shoot off as the network converges, a decreasing value of the learning rate  $\lambda$  is chosen. Empirically the initial value of  $\lambda$  is set to  $10^{-4}$  for first 10 epochs, and then its value is decreased by 5% after each epoch. The decay rate  $\gamma$  is set to a value of 0.9 as recommended in [45]. The numerical stability factor  $\epsilon$  is set to a small value of  $10^{-8}$  in order to avoid division by zero, and the remaining term  $MS$  is initialized to zero.

When a prior information on the weights and biases of a network is available, the network can be trained in a closely related setting using the transfer learning method [47]. In this thesis, the prior information on weights and biases of the proposed network is that of the network trained on different noise settings. For example, if the weights of the network trained for  $\sigma = 10$  and  $p = 0.15$  are available, then the network for  $\sigma = 10$  and  $p = 0.30$  can be trained for this new set of noise parameters by fine-tuning the known weights. While the network takes a large number of epochs to achieve a considerable denoising performance without prior information, it would take only a few epochs to train the network if the initialization considers the known weights. It is also observed that the initial learning rate  $\lambda$  of the networks adopting transfer learning can be lower than that of the networks trained without prior information. In the experiments, the value of  $\lambda$  is set to  $10^{-5}$  for transfer learning.

Experiments are carried out using a computer equipped with an Intel i7 4770K 3.5 GHz processor, 16GB of memory and an NVIDIA Titan Xp GPU. The operating system of the computational environment is Ubuntu 16.04. It takes roughly 6 hours to train each of the networks by running 70 epochs.

Figure 5.1 shows the learning curves of the proposed model in terms of the PSNR obtained by using the validation set for 70 epochs in four different scenarios. The blue solid curve represents the learning curve for the four stage architecture trained for removal of mixed AWGN+SPIN with noise parameters  $\sigma = 10$  and  $p = 0.30$  without any prior information of the weights and biases. The solid red curve with circle markers depicts the learning curve when only the ROF layer is employed, but the upsampling through the  $BI$  layer and the subsequent downsampling through the  $MP_{K_1}$  layer are removed. The pink dotted curve is obtained when both the pre-processing layers, i.e., the ROF and  $BI$  layers are removed. Other parameters remain unchanged in all these scenarios. It can be inferred from Figure 5.1 that the both of the pre-processing layers are essential to obtain a desired level of denoising performance along with the traditional layers of CNN. In addition, the black dashed curve presents the learning curve for the same task, but this time the network is trained by initializing the weights and biases with known data that are obtained from training the network with noise parameters  $\sigma = 10$  and  $p = 0.15$ . It is seen from Figure 5.1 that the denoising performance reaches a satisfactory level very quickly in about 10 epochs in the case of transfer learning, thus reducing the training time significantly.

### 5.4.2 Learned Filters

As per the assigned dimensions of the layers of the CNN model, there are 100, 200, 100 and 1 filters that are obtained for the filter sets  $\mathbf{W}_1$ ,  $\mathbf{W}_2$ ,  $\mathbf{W}_3$  and  $\mathbf{W}_4$ , respectively, when  $\sigma < 20$ . Figure 5.2 shows the entire set of 100 filters of  $\mathbf{W}_1$  learned from the training dataset for SPIN parameters  $\sigma = 10$  and  $p = 0.30$  and used in the first convolution layer. The weights of any of these filters are combinations of positive and negative real values as shown by the grayscale square-shaped dots (see Figure 5.2). These values of weights reveal that each of the filters are bandpass in nature. Similar type of filters also result for  $\mathbf{W}_2$ ,  $\mathbf{W}_3$ ,  $\mathbf{W}_4$ , but are not shown in view of space limitations. These filters produce certain bandpass versions of the noisy image in the output channel of the convolution layers. Figure 5.3 shows the input noisy image, a few output images resulting from the bandpass filters of the first, second, and third convolution layers and the denoised output image resulting from the final or fourth convolution layer. It is

Table 5.1: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 1.

Parameters	Cai's [1]	Cai's [1] +BM3D [2]	WESNR [4]	Proposed IIL [5]
$\sigma = 10$	$27.18 \pm 0.74$	$30.48 \pm 2.11$	$28.40 \pm 2.68$	<b><math>31.28 \pm 2.06</math></b>
$p = 0.15$	$0.7618 \pm 0.0817$	$0.9190 \pm 0.0281$	$0.8669 \pm 0.0457$	<b><math>0.9256 \pm 0.0251</math></b>
$\sigma = 10$	$26.48 \pm 1.17$	$28.76 \pm 2.23$	$27.33 \pm 2.89$	<b><math>29.81 \pm 2.22</math></b>
$p = 0.30$	$0.7572 \pm 0.0701$	$0.8938 \pm 0.0350$	$0.8500 \pm 0.0482$	<b><math>0.9069 \pm 0.0316</math></b>
$\sigma = 15$	$24.41 \pm 0.41$	$29.05 \pm 2.03$	$26.58 \pm 2.63$	<b><math>29.46 \pm 1.97</math></b>
$p = 0.15$	$0.6460 \pm 0.1049$	<b><math>0.8831 \pm 0.0409</math></b>	$0.8047 \pm 0.0751$	$0.8823 \pm 0.0489$
$\sigma = 15$	$24.24 \pm 0.74$	$27.74 \pm 2.13$	$25.74 \pm 2.86$	<b><math>28.40 \pm 2.07</math></b>
$p = 0.30$	$0.6464 \pm 0.0948$	$0.8561 \pm 0.0469$	$0.7904 \pm 0.0762$	<b><math>0.8694 \pm 0.0416</math></b>
$\sigma = 20$	$22.21 \pm 0.26$	$27.85 \pm 2.00$	$25.86 \pm 2.59$	<b><math>28.00 \pm 1.90</math></b>
$p = 0.15$	$0.5536 \pm 0.1127$	$0.8486 \pm 0.0504$	$0.7850 \pm 0.0739$	<b><math>0.8517 \pm 0.0457</math></b>
$\sigma = 20$	$22.32 \pm 0.48$	$26.80 \pm 2.05$	$24.99 \pm 2.85$	<b><math>27.11 \pm 1.97</math></b>
$p = 0.30$	$0.5574 \pm 0.01049$	$0.8208 \pm 0.0553$	$0.7682 \pm 0.0744$	<b><math>0.8297 \pm 0.0553</math></b>
$\sigma = 25$	$20.40 \pm 0.18$	$26.90 \pm 1.98$	$24.87 \pm 2.68$	<b><math>26.93 \pm 1.81</math></b>
$p = 0.15$	$0.4798 \pm 0.1131$	<b><math>0.8173 \pm 0.0580</math></b>	$0.7524 \pm 0.0826$	$0.8151 \pm 0.0566$
$\sigma = 25$	$20.68 \pm 0.35$	$25.99 \pm 2.01$	$23.92 \pm 2.97$	<b><math>26.17 \pm 1.87</math></b>
$p = 0.30$	$0.4854 \pm 0.1071$	$0.7880 \pm 0.0612$	$0.7344 \pm 0.0827$	<b><math>0.7942 \pm 0.0562</math></b>

to be pointed out that in the first three convolution layers, the number of outputs are 100, 200, and 100, respectively. But, due to space limitations, we show only 10 typical output images of each of the two layers. It is seen from Figure 5.3 that the image features such as the textures and edges are extracted selectively in the convolution layers. The images marked as 1, 2, and 3 in the red color are the low resolution versions of the original image, and such an image is propagated through the network independent of the noise settings. Finally, the output of the fourth convolution layer of the network provides the estimate of the noise-free image.

Table 5.2: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 2.

Parameters	Cai's [1]	Cai's [1] +BM3D [2]	WESNR [4]	Proposed IIL [5]
$\sigma = 10$	$27.20 \pm 0.84$	$30.47 \pm 2.13$	$27.80 \pm 2.82$	<b><math>31.29 \pm 2.08</math></b>
$p = 0.15$	$0.7628 \pm 0.0819$	$0.9189 \pm 0.0288$	$0.8440 \pm 0.0693$	<b><math>0.9253 \pm 0.0260</math></b>
$\sigma = 10$	$26.45 \pm 1.20$	$28.74 \pm 2.26$	$26.83 \pm 2.93$	<b><math>29.81 \pm 2.23</math></b>
$p = 0.30$	$0.7564 \pm 0.0721$	$0.8936 \pm 0.0335$	$0.8283 \pm 0.0687$	<b><math>0.9066 \pm 0.0320</math></b>
$\sigma = 15$	$24.43 \pm 0.59$	$29.06 \pm 2.06$	$26.57 \pm 2.64$	<b><math>29.52 \pm 1.98</math></b>
$p = 0.15$	$0.6469 \pm 0.1062$	$0.8831 \pm 0.0413$	$0.8045 \pm 0.0751$	<b><math>0.8866 \pm 0.0402</math></b>
$\sigma = 15$	$24.24 \pm 0.81$	$27.73 \pm 2.12$	$25.74 \pm 3.84$	<b><math>28.41 \pm 2.06</math></b>
$p = 0.30$	$0.6464 \pm 0.0971$	$0.8560 \pm 0.0476$	$0.7903 \pm 0.0761$	<b><math>0.8688 \pm 0.0424</math></b>
$\sigma = 20$	$22.22 \pm 0.52$	$27.86 \pm 2.03$	$25.75 \pm 2.64$	<b><math>28.14 \pm 1.93</math></b>
$p = 0.15$	$0.5543 \pm 0.1141$	$0.8488 \pm 0.0510$	$0.7790 \pm 0.0795$	<b><math>0.8518 \pm 0.0474</math></b>
$\sigma = 20$	$22.33 \pm 0.65$	$26.80 \pm 2.06$	$24.89 \pm 2.89$	<b><math>27.15 \pm 1.99</math></b>
$p = 0.30$	$0.5577 \pm 0.1062$	$0.8207 \pm 0.0564$	$0.7632 \pm 0.0809$	<b><math>0.8291 \pm 0.0566</math></b>
$\sigma = 25$	$20.40 \pm 0.51$	$26.88 \pm 2.00$	$24.85 \pm 2.67$	<b><math>26.95 \pm 1.85</math></b>
$p = 0.15$	$0.4794 \pm 0.1144$	<b><math>0.8169 \pm 0.0589</math></b>	$0.7520 \pm 0.0829$	$0.8131 \pm 0.0602$
$\sigma = 25$	$20.68 \pm 0.60$	$25.98 \pm 2.05$	$23.90 \pm 3.03$	<b><math>26.13 \pm 1.90</math></b>
$p = 0.30$	$0.4855 \pm 0.1071$	$0.7874 \pm 0.0621$	$0.7329 \pm 0.0835$	<b><math>0.7922 \pm 0.0612</math></b>

### 5.4.3 Methods Used for Comparison

The results of the proposed CNN-based method are compared with that of three well established methods when employed to remove mixed-noise. A brief description of the comparing methods are given below:

- Cai's Method [1]: In this method, the impulse noise is detected first as outliers and removed by using AMF. Then, the variational approach-based denoising is employed assuming that the pixels restored in the first phase are essentially free of outliers. The method can also be employed as an ROF for reducing mixed-noise.
- ROF+BM3D [2]: In this method, ROF is first used to remove impulse noise. Then,

Table 5.3: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 3.

Parameters	Cai's [1]	Cai's [1] +BM3D [2]	WESNR [4]	Proposed IIL [5]
$\sigma = 10$	$27.21 \pm 0.83$	$31.02 \pm 2.96$	$27.84 \pm 4.26$	<b><math>31.61 \pm 2.75</math></b>
$p = 0.15$	$0.6794 \pm 0.0856$	$0.8650 \pm 0.0491$	$0.8414 \pm 0.0536$	<b><math>0.9067 \pm 0.0348</math></b>
$\sigma = 10$	$26.53 \pm 1.35$	$29.21 \pm 3.13$	$27.29 \pm 4.79$	<b><math>29.98 \pm 2.55</math></b>
$p = 0.30$	$0.6787 \pm 0.0740$	$0.8788 \pm 0.0418$	$0.8233 \pm 0.0560$	<b><math>0.9116 \pm 0.0329</math></b>
$\sigma = 15$	$24.34 \pm 0.44$	$29.11 \pm 2.32$	$25.73 \pm 3.64$	<b><math>29.48 \pm 2.25</math></b>
$p = 0.15$	$0.5703 \pm 0.0655$	$0.8631 \pm 0.0474$	$0.7744 \pm 0.0852$	<b><math>0.8642 \pm 0.0461</math></b>
$\sigma = 15$	$24.16 \pm 0.76$	$27.73 \pm 2.35$	$24.98 \pm 3.85$	<b><math>28.38 \pm 2.26</math></b>
$p = 0.30$	$0.5731 \pm 0.1236$	$0.8331 \pm 0.0557$	$0.7619 \pm 0.0867$	<b><math>0.8489 \pm 0.0468</math></b>
$\sigma = 20$	$22.20 \pm 0.47$	$28.59 \pm 2.79$	$25.75 \pm 4.09$	<b><math>28.60 \pm 2.42</math></b>
$p = 0.15$	$0.4482 \pm 0.1011$	$0.8322 \pm 0.0569$	$0.7622 \pm \pm 0.0852$	<b><math>0.8234 \pm 0.0540</math></b>
$\sigma = 20$	$22.35 \pm 0.53$	$27.47 \pm 3.83$	$25.02 \pm 4.27$	<b><math>27.66 \pm 2.50</math></b>
$p = 0.30$	$0.4529 \pm 0.0945$	$0.8017 \pm 0.0617$	$0.7439 \pm 0.0894$	<b><math>0.8072 \pm 0.0539</math></b>
$\sigma = 25$	$20.38 \pm 0.18$	$27.11 \pm 2.12$	$24.37 \pm 3.58$	<b><math>27.18 \pm 2.00</math></b>
$p = 0.15$	$0.4009 \pm 0.0658$	<b><math>0.7961 \pm 0.0652</math></b>	$0.7265 \pm 0.0941$	$0.7953 \pm 0.0597$
$\sigma = 25$	$20.66 \pm 0.33$	$26.16 \pm 2.10$	$23.57 \pm 3.65$	<b><math>27.04 \pm 2.01</math></b>
$p = 0.30$	$0.4056 \pm 0.0605$	$0.7641 \pm 0.0673$	$0.7118 \pm 0.0952$	<b><math>0.7845 \pm 0.0610</math></b>

the shape-adaptive self-similar patches from images are grouped and denoised using the 3D collaborative filtering. In tradition, the median filter is employed as an ROF for performance comparison with this method (see, for example, [4]). In our experiments, the Cai's method is chosen as ROF for mixed AWGN+SPIN. On the other hand, for mixed AWGN+SPIN+RVIN, the choice of ROF is AMF followed by ACWMF to remove SPIN and RVIN sequentially.

- WESNR [4]: The method encodes the noisy image patches over a set of principle component analysis-based dictionaries learned offline and weights the coding residuals to suppress the noise. The weighted encoding is performed by integrating the image spar-

Table 5.4: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN+RVIN from Test Sets 1, 2, and 3.

Parameters	Test Sets	AMF+ACWMF +BM3D [2]	WESNR [4]	Proposed IIL [5]
$\sigma = 10$ $p = 0.20$ $r = 0.05$	Test Set 1	$26.52 \pm 2.68$	$26.70 \pm 2.73$	<b><math>26.96 \pm 2.67</math></b>
		$0.8407 \pm 0.0542$	$0.8281 \pm 0.0574$	<b><math>0.8565 \pm 0.0468</math></b>
	Test Set 2	$26.73 \pm 2.73$	$26.49 \pm 2.79$	<b><math>27.18 \pm 2.75</math></b>
		$0.8493 \pm 0.0498$	$0.8156 \pm 0.0720$	<b><math>0.8605 \pm 0.0469</math></b>
	Test Set 3	$26.60 \pm 3.09$	$26.45 \pm 3.18$	<b><math>26.93 \pm 3.07</math></b>
		$0.8254 \pm 0.0631$	$0.8086 \pm 0.0630$	<b><math>0.8365 \pm 0.0568</math></b>
$\sigma = 20$ $p = 0.15$ $r = 0.10$	Test Set 1	$24.87 \pm 2.28$	$24.84 \pm 2.39$	<b><math>25.13 \pm 2.27</math></b>
		$0.7523 \pm 0.0694$	$0.7448 \pm 0.0821$	<b><math>0.7677 \pm 0.0690</math></b>
	Test Set 2	$25.15 \pm 2.32$	$24.98 \pm 2.46$	<b><math>25.33 \pm 2.31</math></b>
		$0.7627 \pm 0.0680$	$0.7495 \pm 0.0850$	<b><math>0.7727 \pm 0.0695</math></b>
	Test Set 3	$24.82 \pm 2.50$	$24.79 \pm 2.70$	<b><math>25.06 \pm 2.48</math></b>
		$0.7235 \pm 0.0802$	$0.7229 \pm 0.0928$	<b><math>0.7377 \pm 0.0822</math></b>

sity and nonlocal self-similarity priors in a variational framework.

The Cai’s, WESNR, and BM3D methods are implemented using the codes distributed by the authors of the respective papers. Default settings prescribed are chosen for the WESNR and BM3D methods. In the Cai’s method, the parameter of out-of-focus kernel is set to zero, since we consider only the noisy version of images. It is the regularization parameter  $\beta$  that determines whether Cai’s method would act as an independent method for removal of mixed-noise or as an ROF. In the experiments,  $\beta$  of this method is chosen in such a way that the overall denoising performance is the best. The codes of the proposed CNN-based denoising method and the test images for which the results are reported are shared in the web-link given in [66].



Table 5.5: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN and AWGN+SPIN+RVIN from commonly-referred five test images.

Test Images	AWGN+SPIN ( $\sigma = 20$ and $p = 0.30$ )			AWGN+SPIN+RVIN ( $\sigma = 20$ , $p = 0.15$ and $r = 0.10$ )		
	Cai's [1] +BM3D [2]	WESNR [4]	Proposed IIL [5]	AMF+ACWMF +BM3D [2]	WESNR [4]	Proposed IIL [5]
<i>Peppers</i>	$29.57 \pm 1.58$	$28.52 \pm 1.73$	<b><math>29.99 \pm 1.38</math></b>	$27.98 \pm 1.69$	$28.25 \pm 1.81$	<b><math>28.37 \pm 1.67</math></b>
	$0.8601 \pm 0.0295$	$0.8462 \pm 0.0232$	<b><math>0.8634 \pm 0.0317</math></b>	$0.8290 \pm 0.0266$	$0.8404 \pm 0.0203$	<b><math>0.8425 \pm 0.0266</math></b>
<i>Goldhill</i>	$28.64 \pm 0.81$	$27.37 \pm 0.93$	<b><math>28.81 \pm 0.69</math></b>	$27.50 \pm 0.79$	$27.30 \pm 0.94$	<b><math>27.61 \pm 0.74</math></b>
	$0.7810 \pm 0.0117$	$0.7088 \pm 0.0119$	<b><math>0.7879 \pm 0.0181</math></b>	$0.7360 \pm 0.0104$	$0.7110 \pm 0.0127$	<b><math>0.7377 \pm 0.0124</math></b>
<i>Boat</i>	$28.92 \pm 1.12$	$26.45 \pm 1.16$	<b><math>28.36 \pm 0.90</math></b>	$26.35 \pm 1.06$	$26.24 \pm 0.79$	<b><math>26.57 \pm 1.01</math></b>
	$0.7894 \pm 0.0121$	$0.7197 \pm 0.0241$	<b><math>0.8024 \pm 0.0131</math></b>	$0.7355 \pm 0.0143$	$0.7360 \pm 0.0103$	<b><math>0.7432 \pm 0.0126</math></b>
<i>Man</i>	$28.08 \pm 1.10$	$26.97 \pm 1.10$	<b><math>28.43 \pm 0.94</math></b>	$26.89 \pm 1.05$	$26.92 \pm 1.06$	<b><math>27.14 \pm 0.97</math></b>
	$0.7958 \pm 0.0027$	$0.7451 \pm 0.0053$	<b><math>0.8075 \pm 0.0085</math></b>	$0.7534 \pm 0.0038$	$0.7450 \pm 0.0042$	<b><math>0.7610 \pm 0.0037</math></b>
<i>Baboon</i>	$25.16 \pm 0.69$	$23.71 \pm 1.08$	<b><math>25.61 \pm 0.63</math></b>	$23.71 \pm 1.11$	$23.65 \pm 1.05$	<b><math>23.87 \pm 1.08</math></b>
	$0.6862 \pm 0.0158$	$0.5569 \pm 0.0217$	<b><math>0.7223 \pm 0.0141</math></b>	$0.5970 \pm 0.0129$	$0.5690 \pm 0.0209$	<b><math>0.6027 \pm 0.0104</math></b>
Average	$27.88 \pm 1.06$	$26.61 \pm 1.2$	<b><math>28.24 \pm 0.91</math></b>	$26.48 \pm 1.16$	$26.47 \pm 1.13$	<b><math>26.71 \pm 1.07</math></b>
	$0.7825 \pm 0.0144$	$0.7154 \pm 0.0172$	<b><math>0.7968 \pm 0.0171</math></b>	$0.7302 \pm 0.0134$	$0.7203 \pm 0.0136$	<b><math>0.7374 \pm 0.0131</math></b>

#### 5.4.4 Results

In the experiments, the denoising performance of the methods are compared in three ways. First, the PSNR and the SSIM indices [58] are used to assess the denoising performance quantitatively. Next, the qualitative evaluation is performed by observing the outputs of the methods, i.e., the denoised images. Finally, the methods are compared in terms of the computational load to denoise an image.

#### Quantitative Evaluation

Tables 5.1, 5.2 and 5.3 show the average denoising performance for different settings of parameters of mixed AWGN+SPIN in Test Sets 1, 2 and 3, respectively, in terms of the metrics PSNR and SSIM for the methods being compared. It is seen from Table 5.1 that the average values of PSNR and SSIM considering the 1000 images in Test Set 1 are the highest among the comparing methods in most of the cases (14 out of 16 instances). For example, the improve-

Table 5.6: Denoising performance in terms of mean and standard deviation of PSNR (in dB) and SSIM for reducing AWGN from Test Set 1.

Parameter	BM3D [2]	WESNR [4]	Proposed IIL [5]
$\sigma = 10$	$33.93 \pm \mathbf{1.81}$	$30.63 \pm 2.57$	$\mathbf{33.50} \pm 1.86$
	$0.9388 \pm 0.0240$	$0.9050 \pm 0.0267$	$\mathbf{0.9438} \pm \mathbf{0.0222}$
$\sigma = 15$	$30.53 \pm \mathbf{1.88}$	$28.00 \pm 2.80$	$\mathbf{31.14} \pm 1.90$
	$0.9034 \pm 0.0361$	$0.8520 \pm 0.0567$	$\mathbf{0.9110} \pm \mathbf{0.0336}$
$\sigma = 20$	$28.93 \pm \mathbf{1.92}$	$26.80 \pm 2.90$	$\mathbf{29.51} \pm 1.93$
	$0.8703 \pm 0.0462$	$0.8236 \pm 0.0551$	$\mathbf{0.8804} \pm \mathbf{0.0443}$
$\sigma = 25$	$27.74 \pm 1.94$	$25.04 \pm 3.30$	$\mathbf{28.31} \pm \mathbf{1.93}$
	$0.8395 \pm 0.0543$	$0.7836 \pm 0.0689$	$\mathbf{0.8509} \pm \mathbf{0.0533}$

ment of the CNN-based method over the closest one, Cai’s+BM3D, is approximately 2.6% in terms of average values of PSNR and it is 0.72% in terms of that of SSIM for  $\sigma = 10$  and  $p = 0.15$ . As the strength of noise is increased, the proposed method consistently performs better than the methods compared in terms of mean of PSNR. It is also observed from the table that the standard deviation of the metric PSNR obtained by the proposed method is always the lowest and that of the metric SSIM is the lowest for 7 out of 8 instances in comparison to the other methods, thus providing the most robust denoising performance. For example, when the noise parameters are  $\sigma = 10$  and  $p = 0.15$ , the gains in robustness over the closest competing method Cai’s+BM3D are 2.4% for PSNR and 10.7% for SSIM. Similarly, the proposed CNN-based method provides the overall best denoising performance in terms of the average of PSNR and SSIM for removal of AWGN+SPIN from 1000 images of Test Set 2 and 15 images of Test Set 3, as observed from Tables 5.2 and 5.3, respectively. In particular, the improvements of the proposed method over Cai’s+BM3D in terms of the performance metrics are observed for 30 and 31 instances out of 32 for the Test Sets 2 and 3, respectively. In summary, it can be inferred from Tables 5.1-5.3 that, in general, the closest competitor of the proposed CNN-based

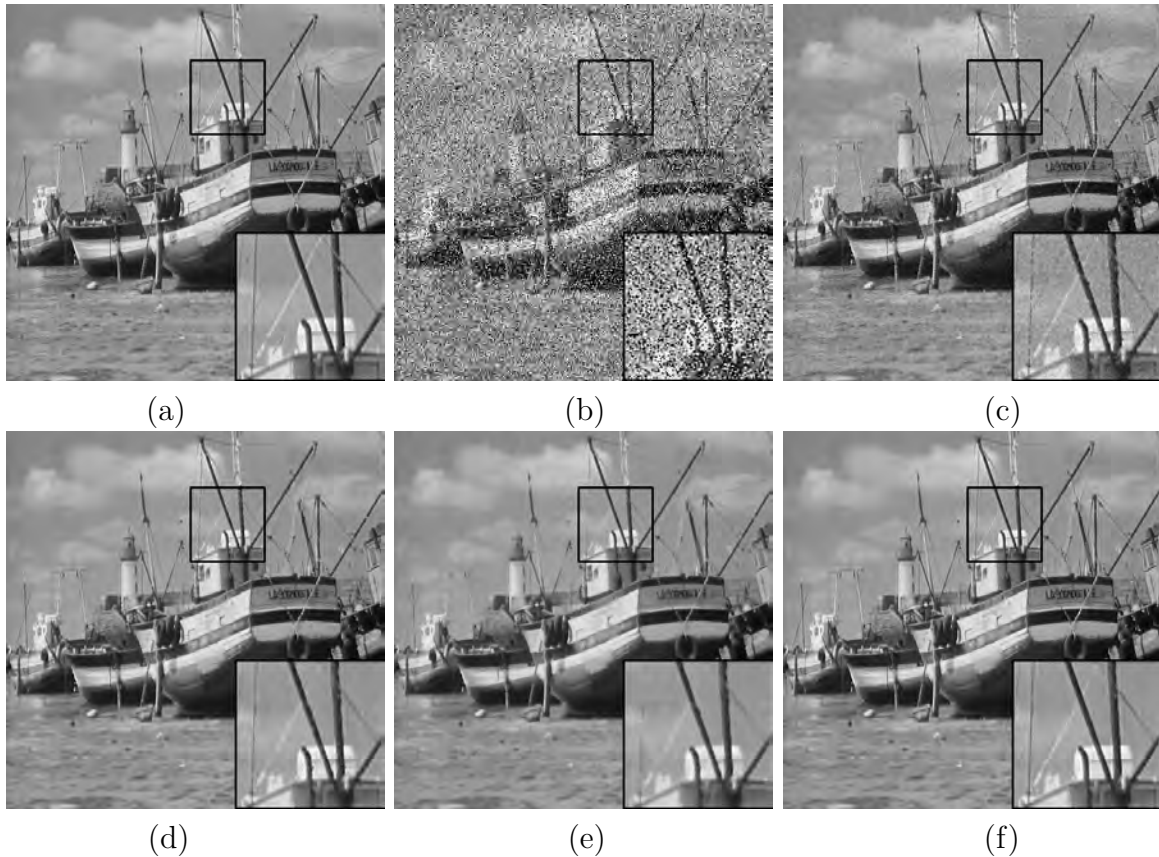


Figure 5.4: Visual comparison of the denoising performance of the methods for the test image *Boat*. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$  (PSNR: 10.65 dB, SSIM: 0.0731). (c) The image is obtained by using Cai's method (PSNR: 27.70 dB, SSIM: 0.6929). The estimated noise-free images are obtained by using the methods (d) Cai's+BM3D [2] (PSNR: 31.21 dB, SSIM: 0.8529), (e) WESNR [4] (PSNR: 29.56 dB, SSIM: 0.8062), and (f) proposed CNN (PSNR: 31.92 dB, SSIM: 0.8596).

method for removal of mixed AWGN+SPIN from the Test Sets is the method Cai's+BM3D.

Table 5.4 presents the comparative performance in terms of the metrics PSNR and SSIM for the challenging case of reducing the mixed AWGN+SPIN+RVIN from Test Sets 1, 2 and 3. It can be observed from the table that the overall performance in terms of the mean and standard deviation of the metrics is superior for the proposed CNN-based method. In the case when the strength of noise is relatively low, i.e.,  $\sigma = 10$ ,  $p = 0.20$  and  $r = 0.05$ , the improvement in average of PSNR is 0.97% and that of SSIM is 1.88% over the competitive method WESNR and AMF+ACWMF+BM3D in Test Set 1. The improvement in robustness

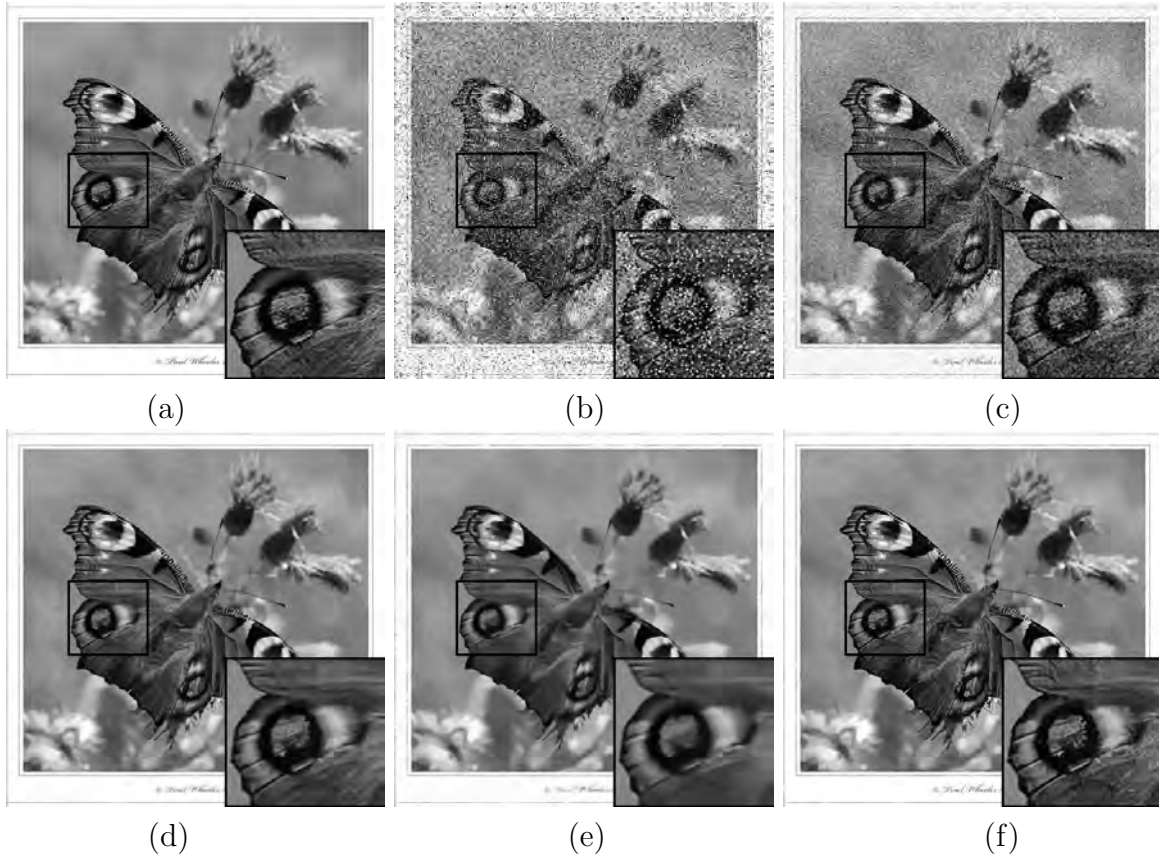


Figure 5.5: Visual comparison of the denoising performance of the methods on a high quality image having a close-up view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 25$  and  $p = 0.15$  (PSNR: 12.40 dB, SSIM: 0.1419). (c) The image is obtained by using Cai's method (PSNR: 20.40 dB, SSIM: 0.3587). The estimated noise-free images are obtained by using the methods (d) Cai's+BM3D [2] (PSNR: 28.71 dB, SSIM: 0.8838), (e) WESNR [4] (PSNR: 25.77 dB, SSIM: 0.8458), and (f) proposed CNN (PSNR: 28.81 dB, SSIM: 0.8906).

in this case is 0.35% in PSNR and 13.7% in SSIM. For a higher level noise with  $\sigma = 20$ ,  $p = 0.15$  and  $r = 0.15$ , the methods AMF+ACWMF+BM3D and WESNR show a performance that is competitive with the proposed method in terms of the metrics of average of PSNR and SSIM, respectively. For example, the improvement in average of PSNR is 1% and that in SSIM is 2% over AMF+ACWMF+BM3D in Test Set 1. In this scenario, the improvements in robustness are also significant with 1.3% in PSNR and 0.57% in SSIM. Similar improvements (14 out of 16 instances) in denoising performance for reducing the AWGN+SPIN+RVIN are also observed for Test Sets 2 and 3.

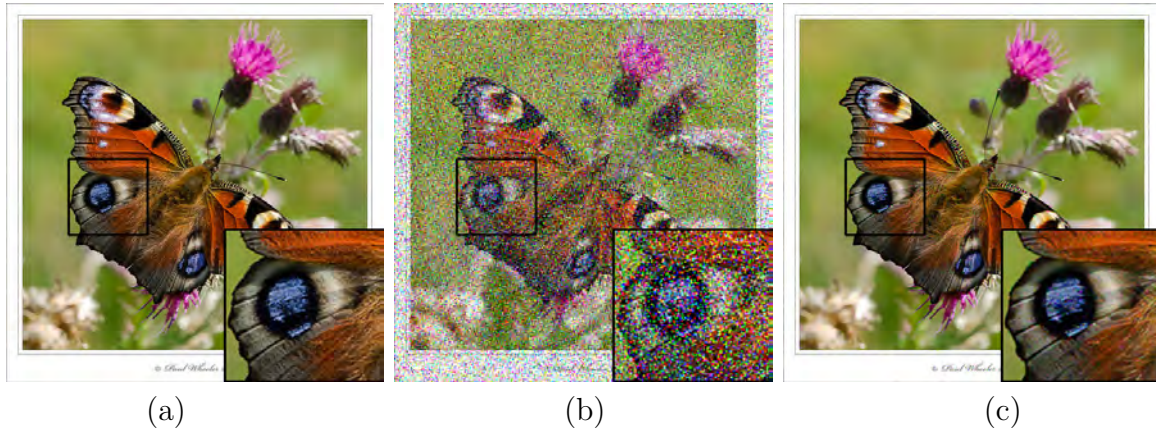


Figure 5.6: Visual comparison of denoising performance of the proposed method for the color version of the high quality image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$  (PSNR: 5.47 dB, SSIM: 0.2466). (c) The estimated noise-free image is obtained by using the proposed CNN (PSNR: 24.79 dB, SSIM: 0.9296).

In order to evaluate the denoising performance of the proposed CNN-based method on commonly-referred test images, we choose five classic grayscale images based on the details in the scene. The images are *Peppers*, *Goldhill*, *Boat*, *Man* and *Baboon*, which have increasing details in the same order. In accordance with the spirit of Test Set 3, the images are considered in three different sizes, namely,  $128 \times 128$ ,  $256 \times 256$  and  $512 \times 512$ . First, the images are corrupted with mixed AWGN+SPIN using the noise parameters  $\sigma = 20$  and  $p = 0.30$ . Then, they are denoised using the methods Cai's+BM3D, WESNR and proposed CNN. This procedure is repeated 5 times and the average results for each of the images are presented in Table 5.5. It is observed from the table that the improvements in terms of PSNR and SSIM resulting from the proposed method over Cai's+BM3D for the relatively smooth image *Peppers* are 1.4% and 0.38%, respectively. The increasing improvement of denoising performance for mixed AWGN+SPIN is observed for the proposed method with the increasing details of images. For example, the improvements in the performance metrics PSNR and SSIM are found to be 1.7% and 5.2%, respectively, for the image *Baboon* that has significant details. It is also evident from the table that standard deviation of the performance metrics is consistently low for the proposed method, thus making it robust denoising method. A similar experiment is conducted for reducing the mixed AWGN+SPIN+RVIN with parameters  $\sigma = 20$ ,  $p = 0.15$  and  $r = 0.10$ ,

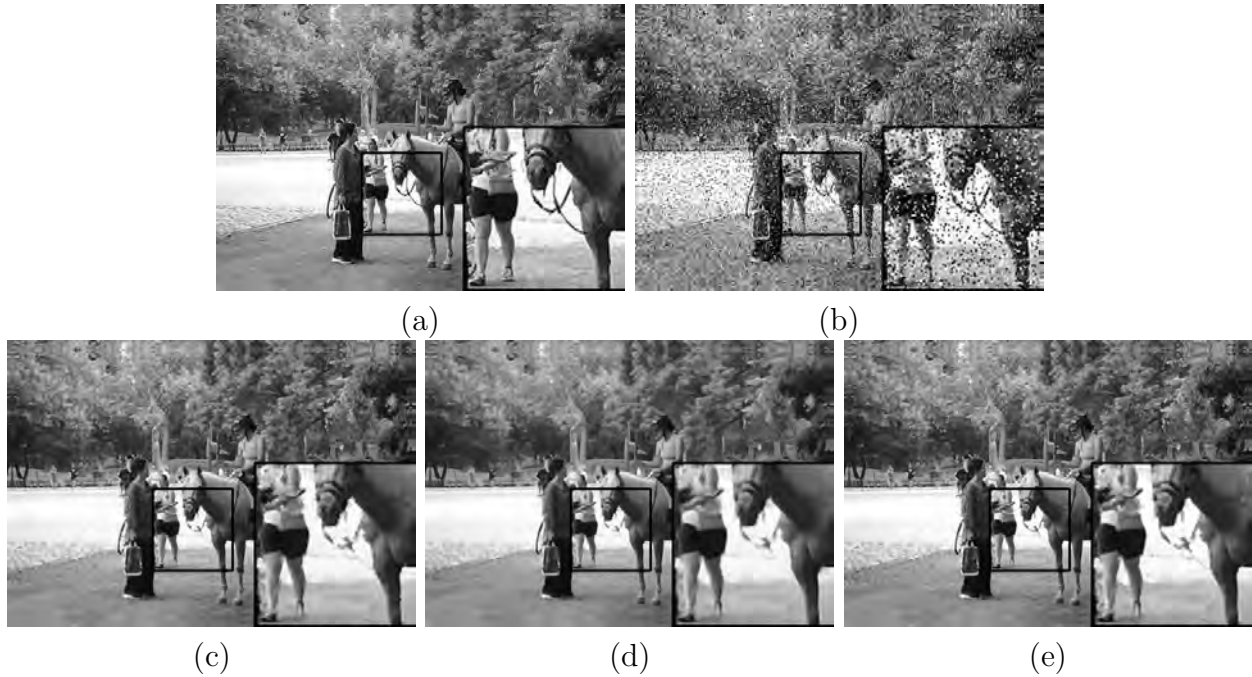


Figure 5.7: Visual comparison of the denoising performance of the methods for mixed AWGN+SPIN+RVIN denoising for an image having long shot view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN+RVIN with parameters  $\sigma = 10$ ,  $p = 0.20$  and  $r = 0.05$  (PSNR: 12.75 dB, SSIM: 0.2316). The estimated noise-free images are obtained by using the methods (c) AMF+ACWMF+BM3D [2] (PSNR: 22.64 dB, SSIM: 0.6893), (d) WESNR [4] (PSNR: 22.44 dB, SSIM: 0.6592), and (e) proposed CNN (PSNR: 22.83 dB, SSIM: 0.7088).

and compared against the methods AMF+ACWMF+BM3D and WESNR. It is also observed that the denoising performance of the proposed method increases with increasing details of the scene for reducing the mixed AWGN+SPIN+RVIN from images.

To verify the effectiveness of the proposed CNN model in reducing AWGN from images, experiments are conducted by excluding the rank order filtering part of the proposed method and that of the WESNR [4]. The performance of the proposed CNN model is also compared with the BM3D method [2], which is well-known for removal of AWGN from images. Table 5.6 presents the denoising performance in terms of PSNR and SSIM for reducing AWGN from Test Set 1. It can be observed from this table that the proposed CNN-based method performs consistently better than the WESNR method. In terms of the mean of the metrics, the proposed method is unbeatable with respect to the BM3D method, but it shows competitive performance

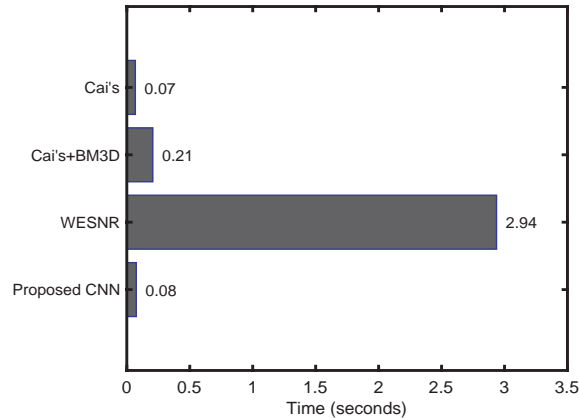


Figure 5.8: Comparison of execution time of the four experimental methods required to denoise an image of size  $128 \times 128$  contaminated by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$ .

in terms of robustness at the low level of noise. Similar results are found for the other test data, but are not included in the thesis to avoid presenting repetitive results.

### Qualitative Evaluation

Examples of commonly-referred test images as well as that of images with close-up and long-shot views are considered for the evaluation of the qualitative performance. Figure 5.4 shows the ground truth of noise-free version of the *Boat* image, its noisy version corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$ , and the denoised images obtained by using the Cai's, Cai's+BM3D, WESNR and proposed CNN-based methods. In this image there are regions with textures such as the 'sands', regions with sharp edges such as the 'masts' of the boats, smooth region such as the 'sky', and text region such as the 'name' of the boat. It can be observed from the denoised images that the proposed CNN reconstructs most of the details such as that of the mast and roof of the engine-room (see zoomed-in region of the image).

Figure 5.5 shows ground truth of noise-free version of a high quality image having a close-up view taken from the Test Set 3, its noisy version corrupted by mixed AWGN+SPIN with parameters  $\sigma = 25$  and  $p = 0.15$ , and the denoised images obtained by using the comparing methods. This image shows a picture of a 'butterfly' in a rectangular 'frame' along with the 'text' of copyright in the bottom. It is seen from Figure 5.5 that the image restored by using

the competitive Cai's+BM3D method shows a good quality edge preservation, when similar edges are present throughout the image such as the frame. For edges and textures that are not repeated, such as the text of copyright or the details of the wings of butterfly, the proposed CNN method clearly provides superior denoising performance. In order to test the performance of the proposed denoising method on the color version of the image, the trained CNN model is used to reduce noise from the red, green and blue channels of the image. Figure 5.6 shows the visual output of the color image, its noisy version corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$ , and the denoised image obtained by the proposed CNN model. It is seen from this figure that the proposed denoising method can successfully reduce mixed-noise from the color version of the image with a performance very similar to that obtained by applying it to the grayscale version.

Figure 5.7 shows a typical image having a long shot view with its noise-free and noisy versions corrupted by mixed AWGN+SPIN+RVIN having parameters  $\sigma = 10$ ,  $p = 0.20$  and  $r = 0.05$ , and the corresponding denoised images obtained by using the AMF+ACWMF+BM3D, WESNR, and proposed CNN methods. Most of the parts of this image have heavy details such as those in 'trees', 'pedestrians', and 'horse' with rider. It can be observed from this figure that the image recovered by using AMF+ACWMF+BM3D has jagged edges (see, for example, leg of 'horse') and that by WESNR is considerably blurred (see, for example, reins of 'horse'). It is seen from the figure that the image recovered by using the proposed CNN-based method resembles original image the most as compared to that recovered by using the method WESNR or AMF+ACWMF+BM3D (see zoomed-in region of the image).

### Evaluation of Computation Time

Experiments are carried out using a computer equipped with an Intel i7 4770K 3.5 GHz processor, 16GB of memory and an NVIDIA Titan Xp GPU. The operating system of the computational environment is Ubuntu 16.04. The three methods being compared, namely, Cai's, WESNR, and ROF+BM3D, are executed in the MATLAB 2016b Linux edition, whereas the proposed CNN is trained in the Tensorflow platform. Figure 5.8 shows a comparative bar-chart representing the mean execution time in seconds required by the four methods for denoising



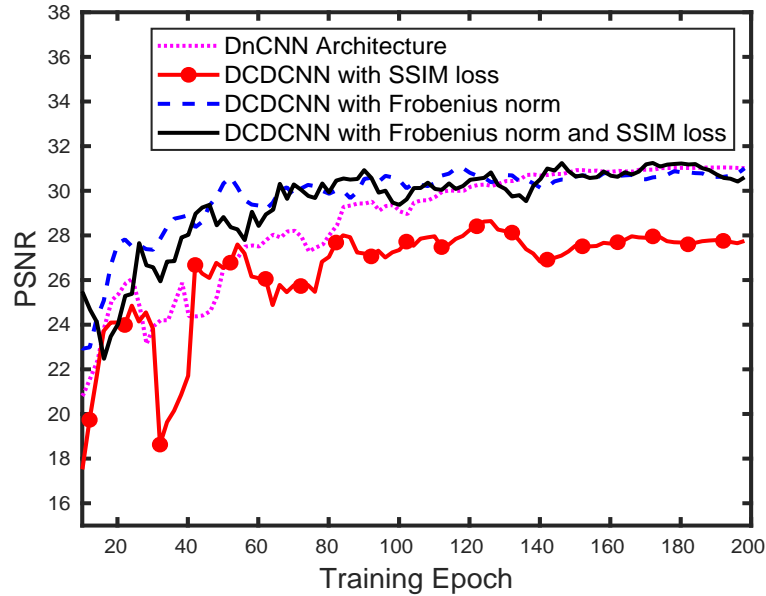


Figure 5.9: Learning curves of the proposed DCDCNN trained for removal of mixed AWGN+SPIN with noise parameters  $\sigma = 25$  and  $p = 0.15$ . The pink dotted learning curve is obtained employing the DnCNN architecture. The solid red curve with circle markers results in DCDCNN architecture is trained employing SSIM loss function. The blue dashed curve is obtained when the DCDCNN architecture is trained using Frobenius norm. The black solid curve is obtained when the DCDCNN is trained employing both the Frobenius norm and SSIM loss function.

an image of size  $128 \times 128$  contaminated by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$ . It can be seen from this figure that the proposed CNN-based method performs the denoising task in less than a fraction of a second. The proposed method is 2.6 times faster than the closest competitor Cai's+BM3D and 36 times faster than the WESNR method. It is to be noted that the AMF+ACWMF requires only few milliseconds over Cai's method and thus is not included in the comparison. Such a small computation time with superior denoising performance puts the proposed CNN-based method to be in a favorable position in comparison to the existing methods.

## 5.5 Experiments on Image to Residual Denoising

### 5.5.1 Model Setup

The values of the parameters of the proposed DCDCNN model described in Section 4.2 are chosen as per the dimensions of the input and output as well as those recommended in practice. Except the convolution layers in the dense blocks and the final convolution layer, the number of filters in first two convolution layers and the transition layer are set to 64 and the kernel size is set to 3 as discussed in 4.2. The number of filter units in each of the dense blocks is set to 6. The number of filters  $C_E$  in each of the  $1 \times 1$  convolution layers in the dense blocks is set to 96 and the number of filters  $C_C$  in each of the  $3 \times 3$  convolution layers in the dense blocks is set to 24. The number of filters in the output  $3 \times 3$  convolution layer is set to number of channels of the input noisy image. For gray level image this value is set to 1. As a result, there are a total of 17  $3 \times 3$  convolution layers and the rest are  $1 \times 1$  convolution layers in the DCDCNN architecture. This arrangement provides a receptive size of  $35 \times 35$ . However, since the dense connections provide a shortcut from input to output in dense blocks, the DCDCNN can learn to employ effective receptive size which may be less than  $35 \times 35$ .

The initialization of the weights and biases of the network and that of the parameters of the training scheme are important for efficient learning. In order to initialize the trainable weights of the convolution filter, He initialization [67] method has been employed where the weights are initialized from a Gaussian distribution and the standard deviation of the distribution is given by

$$\sigma = \sqrt{n_l/2} \quad (5.1)$$

where  $n_l$  is the total number of parameters in a filter set. The bias terms in the convolution layers are initialized to zero. He initialization method is known to avoid vanishing gradient problem during training [67]. In the batch normalization layers the scaling and shifting parameters are set to constant values instead of being trainable and set to 1 and 0 respectively. This is because the scaling and shifting can be learned by the ReLU and convolution layers that

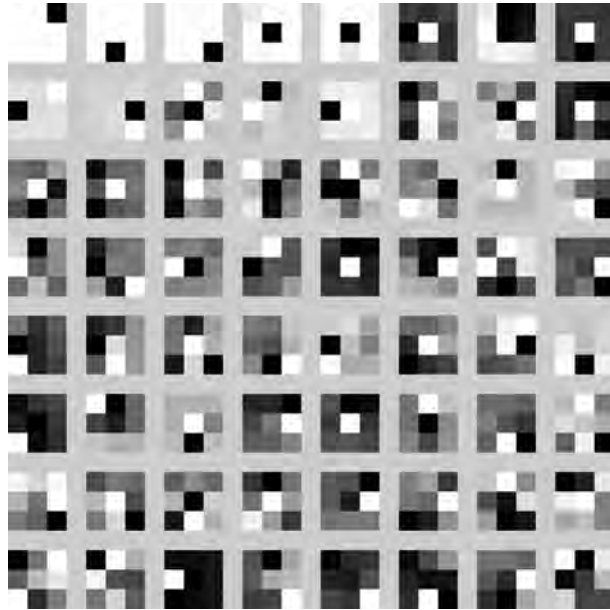


Figure 5.10: Grid of the filters in the filter set of the first convolutions layers of the proposed DCDCNN learned from the training dataset for AWGN+SPIN parameters  $\sigma = 25$  and  $p = 0.15$ . The kernel size of each of the filters is  $3 \times 3$ . The filters are sorted according to the variance of coefficients.

follow the batch normalization layer. The decay rate of the moving average for estimating the population mean and population standard deviation is set to 0.999 as recommended in [51]. The forgetting factor  $\beta_1$  and  $\beta_2$  of the first and second momentum are set to 0.9 and 0.999, as recommended in [59]. The learning rate is empirically set to 0.001 and its value is decreased by 4% after every two epochs. The number of samples in each of the batches during training is set to 15. As stated in Section 4.2, the networks are validated for AWGN denoising for the corresponding  $\sigma$  regardless of the value of  $p$  employed in the training dataset.

Experiments are carried out using a computer equipped with an Intel i7 4770K 3.5 GHz processor, 16GB of memory and an NVIDIA Titan Xp GPU. The operating system of the computational environment is Ubuntu 16.04. It takes roughly a day to train each of the DCDCNN by running 200 epochs. On the other hand, DnCNN takes roughly 16 hours to train on the same training database by running 200 epochs.

Figure 5.9 shows the learning curves of the proposed model in terms of PSNR for mixed AWGN+SPIN removal with noise parameters  $\sigma = 25$  and  $p = 0.15$  obtained by using a

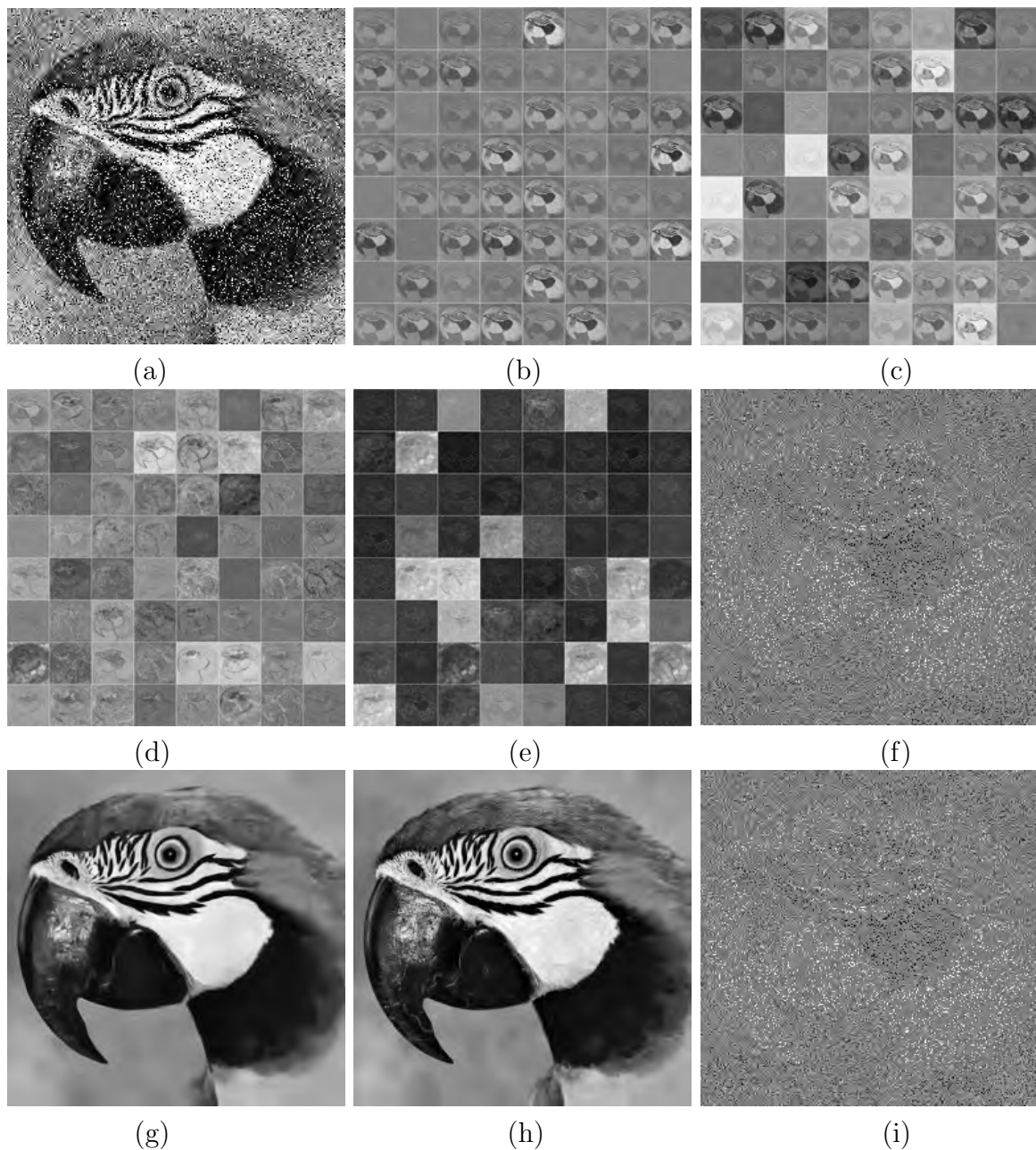


Figure 5.11: The input and outputs of different convolution layers for mixed AWGN+SPIN removal using DCDCNN with noise parameters  $\sigma = 25$  and  $p = 0.15$ . (a) Input noisy images. All 64 outputs of the (b) first and (c) second convolution layers. (d) Typical 64 outputs of the second dense block. (e) The 64 outputs of the final transition layer. (f) Output of the final convolution layer which estimates the residual image. (g) The estimated denoised image (PSNR: 28.76 dB, SSIM: 0.8505). (h) The ground truth image. (i) The ground truth residual image.

validation set of six images contaminated with AWGN noise with parameter  $\sigma = 25$ . The pink dotted learning curve is obtained employing the DnCNN architecture. The solid red curve with circle markers results in DCDCNN architecture is trained employing SSIM loss function. The blue dashed curve is obtained when the DCDCNN architecture is trained using Frobenius norm. The black solid curve is obtained when the DCDCNN is trained employing both the Frobenius norm and SSIM loss function. It can be inferred from Figure 5.9 that the proposed DCDCNN has a faster convergence compared to DnCNN architecture. In terms of loss function, the SSIM loss function, despite designed for ensuring structural similarity, shows a limiting performance during optimization. On the other hand, the DCDCNN employing the Frobenius norm shows the fastest convergence and attains highest validation PSNR of 31.02 dB is obtained at about 118th epoch. When both the Frobenius norm and SSIM loss function are employed, the convergence is still found to be faster than that of DnCNN, however, it becomes slightly slower than the case in which only Frobenius norm is employed in DCDCNN. The highest value of PSNR is found to be 31.24 dB and is obtained at around 146th epoch which shows that employing both the Frobenius norm and SSIM loss function has improved denoising performance. This effect has been observed consistently for other denoising parameters as well.

### 5.5.2 Learned Filters

As per the parameter settings, there are 64 filters in each of the first two convolution layers. Figure 5.10 shows the entire set of 64 filters learned from the training dataset for mixed AWGN+SPIN parameters  $\sigma = 25$  and  $p = 0.15$ . The filters are normalized and have been shown on the grayscale. It is evident that, the first few filters are basically shifting filters which shifts the input images by 1 pixel subject to weighting values of the filter. The other filters are various bandpass filters. The properties of the filters in the other filter sets are similar in nature and hence not shown in the figure.

In order to explore the behavior of these filters the outputs of the filters have been analyzed. Figure 5.11 shows the input noisy image, the outputs of the first two convolution layers, typical 64 outputs of the second dense block, the outputs of the final transition layer, estimated residual image obtained from the final convolution layer, the denoised image, the ground truth image

and the ground truth residual image. It is to be pointed out that, the number of outputs in the second convolution layer is 160, however only typical 64 outputs have been shown in order to avoid clutter. For the same reason, we have not shown the outputs of the other layers. It is evident from the figure that, from input to output of the network, the network progressively removes the image in order to estimate the residual image. The output of the first two layers shows input noisy image at different intensity. However, the output of the latter layers shows only the edges and skeleton of the image from which the residual is estimated. It is also, seen that the denoised image can be efficiently estimated with high visual quality from the estimated residual image.

### 5.5.3 Methods used for Comparison

The results of the proposed CNN-based method are compared with that of three well established methods when employed to remove mixed-noise. A brief description of the comparing methods are given below:

- ROF+BM3D [2]: In this method, ROF is first used to remove impulse noise. Then, the shape-adaptive self-similar patches from images are grouped and denoised using the 3D collaborative filtering. In tradition, the median filter is employed as an ROF for performance comparison with this method (see, for example, [4]). In our experiments, the Cai's method is chosen as ROF for mixed AWGN+SPIN. On the other hand, for mixed AWGN+SPIN+RVIN, the choice of ROF is AMF followed by ACWMF to remove SPIN and RVIN sequentially.
- WESNR [4]: The method encodes the noisy image patches over a set of principle component analysis-based dictionaries learned offline and weights the coding residuals to suppress the noise. The weighted encoding is performed by integrating the image sparsity and nonlocal self-similarity priors in a variational framework.
- Image to Image CNN [5]: In this method, a freed forward CNN is employed which uses image to image learning (IIL) technique where the denoised image is estimated from a

Table 5.7: Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 1.

Parameters	Cai's [1] +BM3D [2]	WESNR [4]	IIL [5]	DnCNN [3]	DCDCNN
$\sigma = 10$	30.47	28.39	31.30	31.63	<b>32.16</b>
$p = 0.15$	0.9190	0.8669	0.9255	0.9213	<b>0.9316</b>
$\sigma = 10$	28.75	27.32	29.82	30.23	<b>30.70</b>
$p = 0.30$	0.8938	0.8500	0.9066	0.9024	<b>0.9126</b>
$\sigma = 25$	26.88	24.86	26.98	27.43	<b>27.73</b>
$p = 0.15$	0.8172	0.7525	0.8130	0.8274	<b>0.8395</b>
$\sigma = 25$	25.98	23.90	26.12	26.63	<b>26.97</b>
$p = 0.30$	0.7874	0.7336	0.7925	0.8039	<b>0.8175</b>
$\sigma = 40$	24.80	21.10	23.31	25.24	<b>25.44</b>
$p = 0.15$	0.7371	0.6311	0.7268	0.7489	<b>0.7583</b>
$\sigma = 40$	24.14	19.46	22.51	24.61	<b>24.92</b>
$p = 0.30$	0.7045	0.5748	0.6971	0.7229	<b>0.7394</b>

pre-processed image. The method pre-processes the image to remove IN by employing ROF and then applies bicubic interpolation to obtain a smooth image. The pre-processed image is then fed to the feed forward CNN which performs further filtering as well as one downsampling operation to obtain the denoised image.

- DnCNN [3]: The method employs a simplest feed forward CNN which uses image to residual learning technique where the residual image is estimated from the input noisy image. The network employs a simple architecture where 17  $3 \times 3$  convolution filters in a row followed by batch normalization and ReLU layers except the output convolution filter.

The Cai's, WESNR and BM3D methods are implemented using the distribution codes provided by the authors of the respective paper. In the Cai's method, out-of-focus kernel in the loss function is set to zero, since we consider only the noisy version of the images. Default values have been employed for other parameters. For WESNR and BM3D methods,

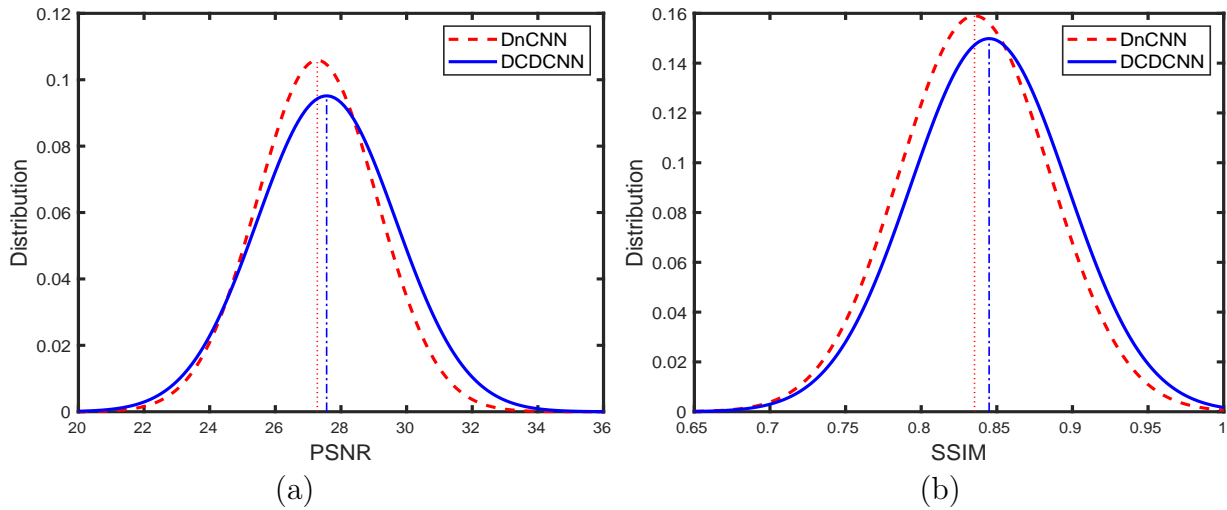


Figure 5.12: (a) Gaussian fit of the distributions of the PSNRs for DnCNN [3] ( $\mu = 27.28$  dB and  $\sigma = 1.82$ ) and DCDCNN ( $\mu = 27.56$  dB and  $\sigma = 2.10$ ). (b) Gaussian fit of the distributions of the SSIMs for DnCNN [3] ( $\mu = 0.8353$  and  $\sigma = 0.0496$ ) and DCDCNN ( $\mu = 0.8449$  and  $\sigma = 0.0518$ ).

the default parameters have been employed. The mixed AWGN-IN removal employing the IIL method and DnCNN architecture are trained employing the same training parameters described in Section 5.5.1.

### 5.5.4 Results

In the experiments, the denoising performance of the methods are compared in three ways. First, the PSNR and the SSIM indices [58] are used to assess the denoising performance quantitatively. Next, the qualitative evaluation is performed by observing the outputs of the methods, i.e., the denoised images. Finally, the methods are compared in terms of the computational load to denoise an image.

#### Quantitative Evaluation

Tables 5.7, 5.8 and 5.9 show the average denoising performance for different settings of parameters of mixed AWGN+SPIN in Test Sets 1, 2 and 3, respectively, in terms of the metrics PSNR and SSIM for the methods being compared. It is seen from Table 5.7 that the average values of PSNR and SSIM considering the 1000 images in Test Set 1 are the highest among the



Table 5.8: Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 2.

Parameters	Cai's [1] +BM3D [2]	WESNR [4]	IIL [5]	DnCNN [3]	DCDCNN
$\sigma = 10$	30.45	27.13	31.28	31.61	<b>32.13</b>
$p = 0.15$	0.9167	0.8179	0.8441	0.9203	<b>0.9307</b>
$\sigma = 10$	28.74	26.30	29.80	30.22	<b>30.69</b>
$p = 0.30$	0.8914	0.8040	0.9069	0.9018	<b>0.9118</b>
$\sigma = 25$	26.93	24.89	26.98	27.39	<b>27.68</b>
$p = 0.15$	0.8191	0.7524	0.8138	0.8242	<b>0.8395</b>
$\sigma = 25$	25.97	23.87	26.10	26.56	<b>26.90</b>
$p = 0.30$	0.7871	0.7316	0.7918	0.7994	<b>0.8138</b>
$\sigma = 40$	24.75	21.10	23.67	25.18	<b>25.37</b>
$p = 0.15$	0.7344	0.6294	0.7221	0.7441	<b>0.7526</b>
$\sigma = 40$	24.12	19.43	22.43	24.55	<b>24.86</b>
$p = 0.30$	0.7033	0.5732	0.6963	0.7168	<b>0.7341</b>

comparing methods in all cases. For example, the improvement of proposed DCDCNN method over the traditional Cai's+BM3D method for  $\sigma = 10$  and  $p = 0.15$  is 5.55%. On the other hand, the improvement of DCDCNN over DnCNN architecture is 1.68%. However, it is seen from the Table 5.7 that the proposed DCDCNN architecture attains highest mean among all of the comparing methods. A natural question arises as whether this high mean is obtained at the expense of robustness. In order to answer the question, the distribution of the proposed method and closest competitor DnCNN are observed and fitted to a Gaussian distribution. Figure 5.12 shows the Gaussian fits of the distributions of PSNRs and SSIMs for DnCNN by using red curve and proposed DCDCNN architecture by using blue curve for denoising mixed AWGN+SPIN for  $\sigma = 25$  and  $p = 0.15$ . The mean of the distributions are marked using red dotted line and blue dash-dotted line for DnCNN and DCDCNN, respectively. It is seen from the figures that, even though the distribution is spreaded for proposed DCDCNN, the mean has a higher value and the values of PSNR and SSIM are higher for almost all of the test cases.

Table 5.9: Denoising performance in terms of mean PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN from Test Set 3.

Parameters	Cai's [1] +BM3D [2]	WESNR [4]	IIL [5]	DnCNN [3]	DCDCNN
$\sigma = 10$	31.01	27.86	31.62	31.87	<b>32.48</b>
$p = 0.15$	0.9063	0.8417	0.9068	0.8994	<b>0.9169</b>
$\sigma = 10$	29.23	26.88	30.01	30.41	<b>30.96</b>
$p = 0.30$	0.8785	0.8232	0.9114	0.8761	<b>0.8858</b>
$\sigma = 25$	27.69	25.00	27.68	28.27	<b>28.60</b>
$p = 0.15$	0.8017	0.7339	0.8011	0.8078	<b>0.8227</b>
$\sigma = 25$	26.72	24.13	27.14	27.43	<b>27.81</b>
$p = 0.30$	0.7705	0.7141	0.7851	0.7886	<b>0.7972</b>
$\sigma = 40$	25.75	22.06	23.91	26.24	<b>26.41</b>
$p = 0.15$	0.7281	0.6305	0.7110	0.7376	<b>0.7392</b>
$\sigma = 40$	24.99	20.67	22.15	25.60	<b>25.97</b>
$p = 0.30$	0.6888	0.5804	0.6754	0.7070	<b>0.7274</b>

This observation is made for other noise parameters as well. Similarly, the proposed DCDCNN provides the overall best denoising performance in terms of the average of PSNR and SSIM for removal of AWGN+SPIN from 1000 images of Test Set 2 and 15 images of Test Set 3, as observed from Tables 5.8 and 5.9, respectively. In summary, it can be inferred from Tables 5.7-5.9 that, in general, the closest competitor of the proposed CNN-based method for removal of mixed AWGN+SPIN from the Test Sets is the method DnCNN for CNN architecture and Cai's+BM3D for traditional method.

In order to evaluate the denoising performance of the proposed DCDCNN on commonly-referred test images, we choose 11 common grayscale images that are widely used in the literature [3]. The images are, *Cameraman*, *House*, *Pepper*, *Starfish*, *Butterfly*, *F21*, *Parrot*, *Lena*, *Boat*, *Man*, and *Couple*. The images are corrupted with mixed AWGN+SPIN using noise parameters  $\sigma = 25$  and  $p = 0.15$ . Then, they are denoised employing the Cai's+BM3D, WESNR, IIL, DnCNN and the proposed DCDCNN. The procedure is repeated five times and the average

Table 5.10: Denoising performance in terms of mean of PSNR (in dB) and SSIM for reducing mixed AWGN+SPIN with  $\sigma = 25$  and  $p = 0.15$  from commonly-referred test images.

Test Images	Cai's [1] +BM3D [2]	WESNR [4]	IIL [5]	DnCNN [3]	DCDCNN
<i>Cameraman</i>	28.24	25.49	28.33	28.93	<b>29.35</b>
	0.8341	0.7967	0.8250	0.8453	<b>0.8604</b>
<i>House</i>	32.24	31.23	31.72	31.91	<b>32.67</b>
	0.8466	0.8368	0.8413	0.8327	<b>0.8557</b>
<i>Pepper</i>	29.10	27.88	29.30	29.65	<b>30.04</b>
	0.8535	0.8361	0.8505	0.8560	<b>0.8696</b>
<i>Starfish</i>	27.78	26.49	27.71	28.33	<b>28.76</b>
	0.8334	0.7909	0.8263	0.8422	<b>0.8542</b>
<i>Butterfly</i>	28.49	26.76	28.67	29.27	<b>29.65</b>
	0.8896	0.8614	0.8905	0.8948	<b>0.9084</b>
<i>F21</i>	27.40	25.62	27.61	28.01	<b>28.29</b>
	0.8357	0.8062	0.8392	0.8377	<b>0.8571</b>
<i>Parrot</i>	27.80	23.61	27.75	28.56	<b>28.77</b>
	0.8326	0.7976	0.8268	0.8411	<b>0.8503</b>
<i>Lena</i>	31.41	30.54	31.03	31.45	<b>31.90</b>
	0.8464	0.8313	0.8402	0.8446	<b>0.8593</b>
<i>Boat</i>	29.24	28.00	29.07	29.35	<b>29.66</b>
	0.7847	0.7334	0.7728	0.7820	<b>0.7941</b>
<i>Man</i>	29.06	28.07	28.95	29.36	<b>29.55</b>
	0.7868	0.7438	0.7790	0.7936	<b>0.8040</b>
<i>Couple</i>	29.06	27.59	28.93	29.07	<b>29.45</b>
	0.8012	0.7375	0.7925	0.7934	<b>0.8101</b>
<i>Average</i>	29.07	27.39	29.01	29.44	<b>29.83</b>
	0.8313	0.7974	0.8258	0.8330	<b>0.8476</b>

results are presented in Table 5.10. It is observed from the table that the mean PSNR and SSIM is highest for the proposed method. For the *Cameraman* image the proposed DCDCNN shows 1.4% improvement in terms of PSNR and 1.8% improvement in terms of SSIM over DnCNN. For a highly detailed image, for example *Butterfly*, the improvement over DnCNN in terms of PSNR and SSIM is 1.3% and 1.5%, respectively. On an average, the improvement in this test set is 1.3% and 1.6% in terms of PSNR and SSIM, respectively.

Table 5.11: Denoising performance in terms of mean of PSNR (in dB) and SSIM for reducing AWGN from commonly referred test images.

Parameter	BM3D [2]	WESNR [4]	IIL [5]	DnCNN [3]	DCDCNN
$\sigma = 10$	34.39	32.88	34.57	34.79	<b>34.80</b>
	0.9235	0.8973	0.9239	0.9276	<b>0.9278</b>
$\sigma = 25$	29.94	28.04	30.08	30.41	<b>30.43</b>
	0.8500	0.8181	0.8489	0.8616	<b>0.8621</b>
$\sigma = 40$	27.61	18.10	27.41	28.16	<b>28.19</b>
	0.7928	0.4970	0.7722	0.8096	<b>0.8105</b>

In order to verify the efficacy of the proposed DCDCNN model in reducing AWGN from images, experiments are conducted on commonly referred 11 test images and compared against well known AWGN denoising method BM3D [2], mixed noise denoising method WESNR [4], and CNN based approach from IIL [5] and DnCNN [3]. The the rank order filtering part of the WESNR, and IIL methods are excluded since they are employed only for impulse reduction. Table 5.11 shows the experimental results. It is seen form the table that the proposed DCDCNN method performs consistently better than the comparing methods. despite removal of AWGN only is a relatively easier problem, the proposed architecture performs better than the closest competitor DnCNN. Similar results are found for the other test sets and are not included in the results to avoid repeating similar results.

### Qualitative Evaluation

In order to qualitatively evaluate the denoising performance of the proposed DCDCNN, images with various types of details at various noise level have been considered. Figure 5.13 shows ground truth of a highly detailed image of a ‘butterfly’ with a close view, its noisy version corrupted by mixed AWGN+SPIN with parameters  $\sigma = 40$  and  $p = 0.15$ , and the denoised images obtained by Cai’s+BM3D, IIL, DnCNN and proposed DCDCNN. The image contains a butterfly within a frame with straight and sharp edges and cursive copyright text at the bottom. It is seen from Figure 5.13 that Cai’s+BM3D provides a good quality edge preservation in the

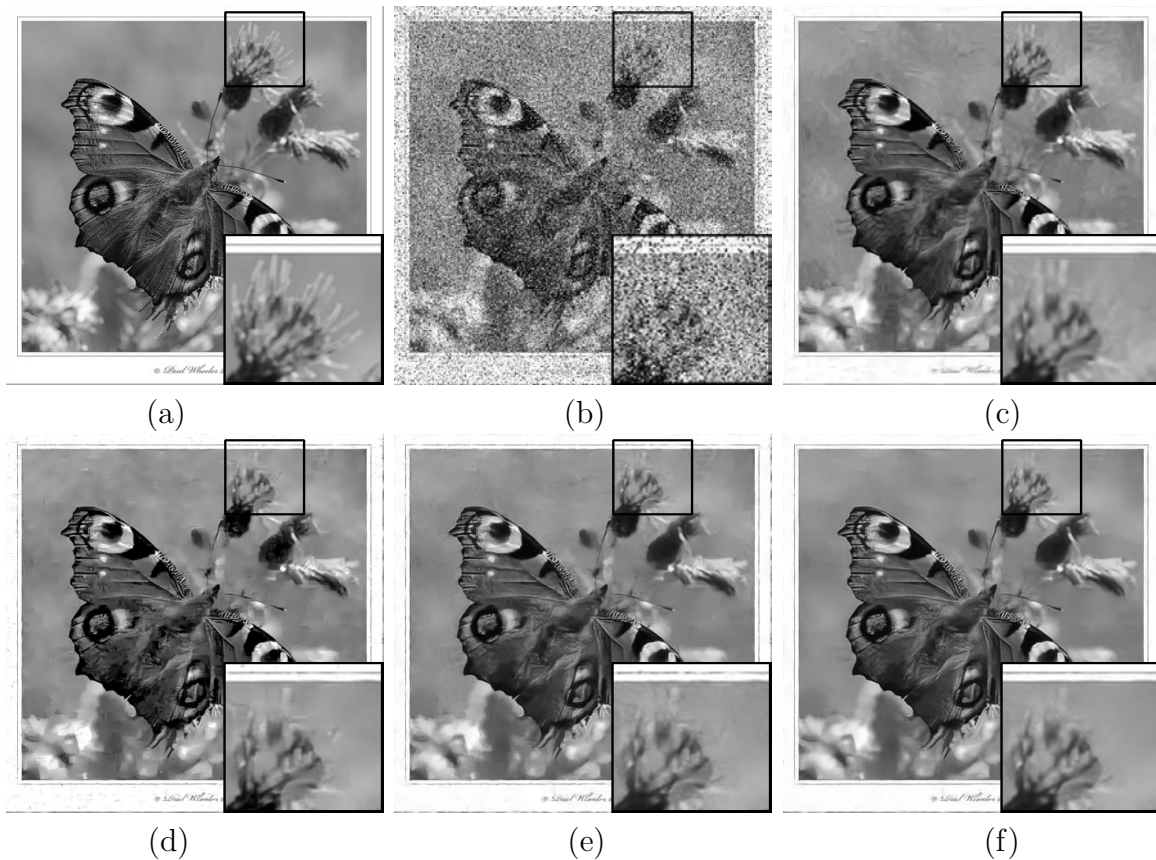


Figure 5.13: Visual comparison of the denoising performance under heavy noise of the methods for a high quality image having a close-up view. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 40$  and  $p = 0.15$  (PSNR: 11.72 dB, SSIM: 0.1226). The estimated noise-free images are obtained by using the methods (c) Cai's [1]+BM3D [2] (PSNR: 26.81 dB, SSIM: 0.8269), (d) IIL [5] (PSNR: 24.50 dB, SSIM: 0.7705), (e) DnCNN [3] (PSNR: 26.73 dB, SSIM: 0.8282), and (f) proposed DCDCNN (PSNR: 27.45 dB, SSIM: 0.8612).

frames as these edges are repetitive structure in the image. The IIL method shows a good edge preservation in the 'frames' portion of the image, however, the image is choppy. The DnCNN method shows a good texture preservation. Overall, the proposed DCDCNN provides a better texture as well as edge preservation and provides a smooth textures in blurred regions. The proposed denoising method has been employed to denoise the color version of the image as well. In this scheme the proposed DCDCNN is employed to reduce noise from the red, green and blue channels of the image. In a similar fashion the IIL method and DnCNN architecture has also been employed to denoise the color image. Figure 5.14 shows the visual output of the color

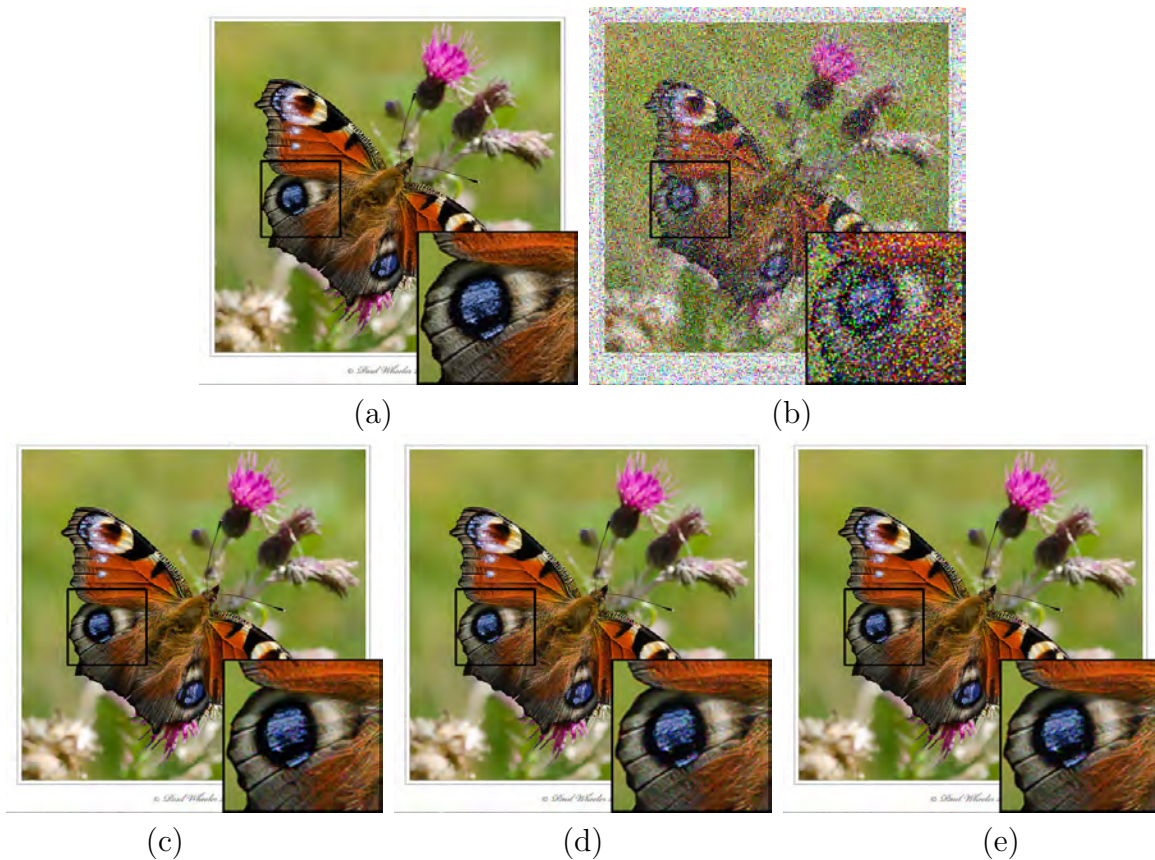


Figure 5.14: Visual comparison of the denoising performance for the color version of the high quality image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 0.30$  (PSNR: 9.86 dB, SSIM: 0.2245). The estimated noise-free images are obtained by using the methods (d) IIL [5] (PSNR: 30.06 dB, SSIM: 0.9686), (e) DnCNN [3] (PSNR: 30.71 dB, SSIM: 0.9725), and (f) proposed DCDCNN (PSNR: 31.5015 dB, SSIM: 0.9757).

image, its noisy version corrupted by mixed AWGN+SPIN with parameters  $\sigma = 10$  and  $p = 30$ , and the denoised image obtained by IIL, DnCNN and DCDCNN. It is seen from the images that, the methods can reduce noise from the color version of images with a performance similar to that obtained by applying it to grayscale version and the proposed DCDCNN provides the best visual output among the three compared methods.

Figure 5.15 show a commonly referred *Parrot* image, its noisy version corrupted by mixed AWGN+SPIN with noise parameter  $\sigma = 25$  and  $p = 0.15$ , and the denoised images obtained by Cai's+BM3D, IIL, DnCNN and proposed DCDCNN. The image has details in the round edges and small feathers. It is seen from the figure that the Cai's+BM3D method provides a

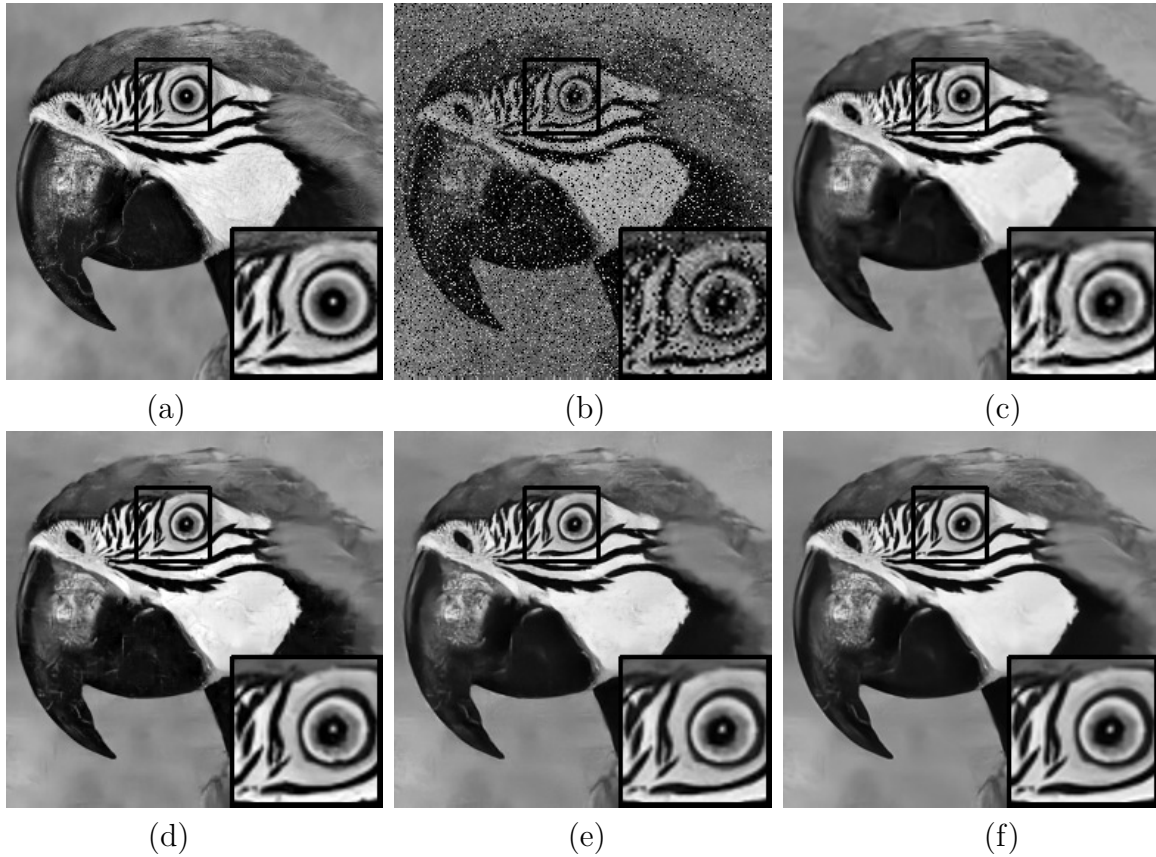


Figure 5.15: Visual comparison of the denoising performance of the methods for commonly referred *Parrot* image. (a) The ground truth of noise-free image. (b) The image is corrupted by mixed AWGN+SPIN with parameters  $\sigma = 25$  and  $p = 0.15$  (PSNR: 12.30 dB, SSIM: 0.1201). The estimated noise-free images are obtained by using the methods (c) Cai's [1]+BM3D [2] (PSNR: 27.80 dB, SSIM: 0.8326), (d) IIL [5] (PSNR: 27.75 dB, SSIM: 0.8268), (e) DnCNN [3] (PSNR: 28.56 dB, SSIM: 0.8411), and (f) proposed DCDCNN (PSNR: 28.77 dB, SSIM: 0.8503).

smooth image. All the CNN based methods provide a better recovery of round edges (see the zoomed in region of the image). However, the proposed DCDCNN provides a superior denoised image without any artifacts near the 'peck' or above the 'head' of the parrot.

In order to evaluate the denoising performance of the proposed DCDCNN on a real image, we chose near-infrared (NIR) image, which can be modeled as mixed AWGN+SPIN [13], of hand from the Technocampus database [68]. The image is used for biometric recognition from the vein pattern. However, due to the appearance of the mixed AWGN-IN noise in the NIR image, the vein pattern cannot be extracted without denoising. Traditionally, median filtering followed by AWGN denoising is applied to remove the noise. In this paper, denoising

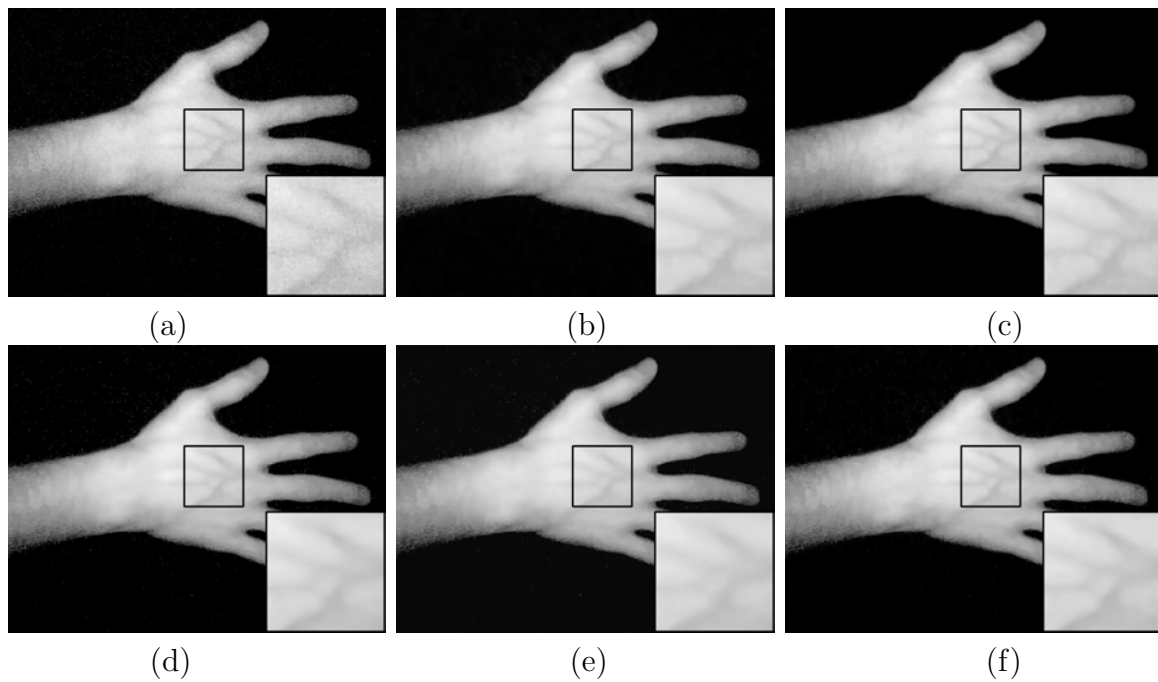


Figure 5.16: Visual comparison of the denoising performance of the methods for a near infrared (NIR) image. (a) The original noisy image. The image is denoised assuming noise parameters  $\sigma = 20$  and  $p = 0.15$  by (b) Cai's [1]+BM3D [2], (c) WESNR [4], (d) IIL [5], (e) DnCNN [3], and (f) proposed DCDCNN.

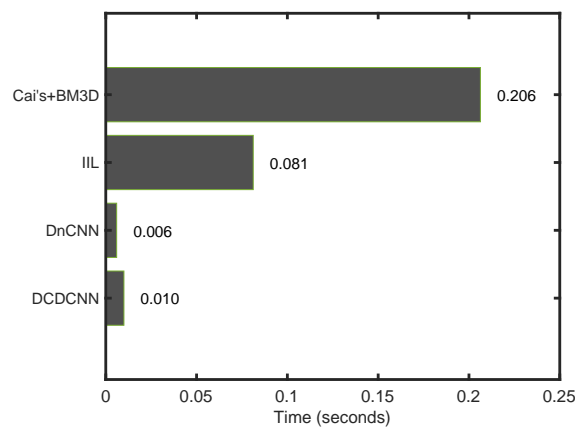


Figure 5.17: Comparison of execution time of the four experimental methods required to denoise an image of size  $128 \times 128$  contaminated by mixed AWGN+SPIN with parameters  $\sigma = 25$  and  $p = 0.15$ .

of NIR images have been performed by the comparing methods. Figure 5.16 shows denoising performance of the methods Cai's+BM3D, WESNR, IIL, DnCNN and proposed DCDCNN on an NIR image. It can be seen from the figure that, only the proposed DCDCNN provides noise



free output with clear vein pattern. The closest method Cai's+Bm3D provides a clear vein pattern however there are smudges and rough patches. The other methods either still contain grains or cause excessive blurring.

### Evaluation of Computation Time

Experiments are carried out using a computer equipped with an Intel i7 4770K 3.5 GHz processor, 16GB of memory and an NVIDIA Titan Xp GPU. The operating system of the computational environment is Ubuntu 16.04. The Cai's method, BM3D and WESNR methods are executed in the MATLAB 2016b Linux edition, whereas the IIL, DnCNN and the proposed DCDCNN are trained in the Tensorflow platform and are executed in GPU. Figure 5.17 shows the mean execution time in seconds required by the four methods, namely, Cai's+BM3D, IIL, DnCNN and proposed DCDCNN, for denoising an image of size  $128 \times 128$  contaminated by mixed AWGN+SPIN with parameters  $\sigma = 25$  and  $p = 0.15$ . The WESNR method requires more than two seconds to denoise each of the images and thus is skipped in the figure. It can be seen from the figure that the CNN based methods perform the denoising task in few milliseconds. The proposed DCDCNN is 2 times slower than the closest architecture DnCNN, as DCDCNN is much deeper and complex network architecture than the DnCNN. However, DCDCNN is 170 times faster than the traditional Cai's+BM3D method. Such a small computation time with superior denoising performance puts the proposed CNN-based method to be in a favorable position in comparison to the existing methods.

# Chapter 6

## Conclusions

In this thesis, CNN-based deep learning approach has been proposed for reduction of mixed Gaussian-impulse noise from images. Restoration of images contaminated by the AWGN and IN is still a challenge even for the handcrafted algorithms due to their contrasting natures. In this context, two learning-based approaches, i.e., image to image learning (IIL) and image to residual learning (IRL), have been adopted such that the necessary restoration process of mixed Gaussian-impulse noise can be developed through extensive training with large number of image variabilities. Among the different approaches of machine learning algorithms, CNN is particularly suitable for image processing and has shown vast representation capabilities. The architecture of CNN is such that the 2D structure of local neighboring regions of images remains unaltered as opposed to the vectorization of image or patch matrices employed in the sparse coding-based methods. Further, the possibility of utilization of an effective number of self-similar patches gathered from huge size training set through a gradient descent learning in the CNN-based algorithm is much higher as compared to that in the case of traditional non-local means or patch-based methods, which collect a sub-optimal set of patches from the test images. In the proposed IIL method, the image corrupted by mixed Gaussian-impulse noise has been preprocessed by rank order filtering and upsampled by means of BI. The filtered and interpolated image is fed to a 4-stage CNN architecture, wherein each stage consists of a suitable set of layers including the convolution, ReLU and max-pooling layers. The dimensions of the four sets of convolution filters and the corresponding bias terms are chosen as per the size of the

input and the output of the interconnected layers and that of the local neighboring mask used in each of the layers. The weights of the filters and biases in the four stages of CNN have been learned from a sufficiently large dataset with the adoption of suitable data augmentation techniques. Experimentation on the validation set of images has revealed that the use of BI layer, which acts as a low-level smoother in the preprocessing stage, significantly improves the overall performance of the CNN-based IIL denoising scheme. A faster training is achieved by adopting the mechanism of ‘transfer learning’, wherein the known weights and biases of a network for a particular setting of noise parameters are used to initialize the same for training the network in a new setting of noise parameters. The proposed method has been extensively tested against well established methods, namely, Cai’s method, ROF+BM3D, and WESNR using three different test datasets and commonly-used individual test images considering different practical scenarios. In particular, not only different imaging environments of the training and testing sets have been considered in the experiments, but also denoising experiments on different settings of mixed-noise scenarios including AWGN+SPIN and AWGN+SPIN+RVIN have been carried out. The strengths of the Gaussian and impulse noise and their mixing proportion have been varied widely to evaluate the effectiveness of the proposed CNN-based model for reducing the mixed-noise. In addition, the effect due to deviations of noise parameters and that of the sizes of images during training and testing phases have been examined. It has been shown that the average values of the commonly-used performance metrics, namely, the PSNR and SSIM for reducing mixed AWGN+SPIN or AWGN+SPIN+RVIN are the highest in almost all cases for the proposed IIL CNN-based method among all the methods compared. The overall robustness of the denoising performance of the proposed method has been found to be significantly superior in terms of these metrics. The visualization of denoised images has also revealed that the proposed method retains the image structure and details maximally as compared to the other methods both in reducing mixed AWGN+SPIN and AWGN+SPIN+RVIN. In the IRL CNN-based denoising scheme, a very deep densely connected convolutional neural network (DCDCNN) has been proposed. Due to its nature of producing intermediate outputs with variable receptive size, the network can optimally denoise the images. Experiments are carried out in a similar fashion to IIL CNN based denoising scheme and it is seen that the DCDCNN

can denoise images with superior performance compared to traditional patch based and sparse representation based methods as well as other CNN based methods in terms of PSNR and SSIM for a variety of noise parameters. Inspection of visual quality by observing the denoising performance on some heavily detailed images show that the proposed DCDCNN produce high quality denoised images even under heavy noise. The experiments carried out on color image shows that the proposed DCDCNN can provide superior denoising performance compared to the other CNN based methods. In addition, it has been observed that the proposed methods require a very insignificant computational time for denoising an image. In conclusion, the light weight structure of the proposed CNN model can play a significant role in many applications, where the low-complexity and robust denoising performance are primary concerns.

## 6.1 Future Works

There are a number of future direction which can be explored to further enrich the study. First of all, there a number of optimization techniques and a number of loss functions can be explored in order to improve the denoising performance further. Additionally, different architectures and layering arrangements can be studied in order to find the optimum architecture for denoising.

Additionally, adversarial examples can compromise the performance of a neural network [69, 70]. Thus a potential study is to make the performance of these neural networks more robust to adversarial examples. A initial approach can be to train the network along with adversarial examples. A challenging task will be to employ an adversarial image detector and take corrective measures to detect fraud and improve the image quality.

Moreover, these neural networks and approach can be employed in other closely related applications, such as, image inpainting and watermark removal. Image super-resolution is another application which can be improved by exploring different architectures along with improvement by employing loss functions and optimization techniques.

# Appendix

The images of Test Set 1 are shown using collage.





The images of Test Set 3 are shown below.



# References

- [1] Jian-Feng Cai, Raymond H. Chan, and Mila Nikolova. Fast two-phase image deblurring under impulse noise. *J. Mathematical Imaging and Vision*, 36(1):46–53, 2010.
- [2] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. Image Processing*, 16(8):2080–2095, 2007.
- [3] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Processing*, 26(7):3142–3155, 2017.
- [4] Jieli Jiang, Lei Zhang, and Jian Yang. Mixed noise removal by weighted encoding with sparse nonlocal regularization. *IEEE Trans. Image Processing*, 23(6):2651–2662, 2014.
- [5] Mohammad Tariqul Islam, S M Mahbubur Rahman, M Omair Ahmad, and M N S Swamy. Mixed Gaussian-impulse noise reduction from images using convolutional neural network. *Signal Processing: Image Communication*, 68:26–41, 2018.
- [6] Tamer Rabie. Adaptive hybrid mean and median filtering of high-ISO long-exposure sensor noise for digital photography. *J. Electronic Imaging*, 13(2):264–277, 2004.
- [7] Chuan Qin, Chin-Chen Chang, and Yi-Ping Chiu. A novel joint data-hiding and compression scheme based on SMVQ and image inpainting. *IEEE Trans. Image Processing*, 23(3):969–978, 2014.



- 
- [8] Chuan Qin, Zhihong He, Heng Yao, Fang Cao, and Liping Gao. Visible watermark removal scheme based on reversible data hiding and image inpainting. *Signal Processing: Image Communication*, 60:160–172, 2018.
- [9] Rafael C Gonzalez and Richard E Woods. *Digital Image Processing*. Prentice Hall, NJ, 3rd edition, 2008.
- [10] Robert Grou-Szabo and Tadashi Shibata. Random-valued impulse noise detector for switching median filters using edge detectors. In *Proc. 3rd Int. Conf. Signal Processing and Communication Systems*, pages 1–4, Omaha, NE, 2009.
- [11] Yoko Norose, Koichi Mizutani, Naoto Wakatsuki, and Tadashi Ebihara. Noise reduction in ultrasonic computerized tomography by preprocessing for projection data. *Japanese Journal of Applied Physics*, 54(7S1):07HC12–1–4, 2015.
- [12] Rastislav Lukac, Konstantinos N Plataniotis, Bogdan Smolka, and Anastasios N Venetianopoulos. A multichannel order-statistic technique for cDNA microarray image processing. *IEEE Trans. Nanobioscience*, 3(4):272–285, 2004.
- [13] Behnood Rasti, Paul Scheunders, Pedram Ghamisi, Giorgio Licciardi, and Jocelyn Chanussot. Noise reduction in hyperspectral imagery: Overview and application. *Remote Sensing*, 10(3):482, 2018.
- [14] G Deng and LW Cahill. An adaptive Gaussian filter for noise reduction and edge detection. In *Proc. IEEE Int. Conf. Record. Nuclear Science Symposium and Medical Imaging Conference*, pages 1615–1619, 1993.
- [15] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, volume 2, pages 60–65, San Deigo, CA, 2005.
- [16] Hangfan Liu, Ruiqin Xiong, Jian Zhang, and Wen Gao. Image denoising via adaptive soft-thresholding based on non-local samples. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 484–492, Boston, MA, 2015.

- 
- [17] David L Donoho. De-noising by soft-thresholding. *IEEE Trans. Information Theory*, 41(3):613–627, 1995.
- [18] S. M. Mahbubur Rahman and Md. Kamrul Hasan. Wavelet-domain iterative center weighted median filter for image denoising. *Signal Processing*, 83(5):1001–1012, 2003.
- [19] S. M. Mahbubur Rahman, M. Omair Ahmad, and M. N. S. Swamy. Bayesian wavelet-based image denoising using the Gauss–Hermite expansion. *IEEE Trans. Image Processing*, 17(10):1755–1771, 2008.
- [20] Jing Liu, Yinghui Wang, Kaijun Su, and Wenjuan He. Image denoising with multidirectional shrinkage in directionlet domain. *Signal Processing*, 125:64–78, 2016.
- [21] Hamidreza Sadreazami, M Omair Ahmad, and MNS Swamy. A study on image denoising in contourlet domain using the alpha-stable family of distributions. *Signal Processing*, 128:459–473, 2016.
- [22] Jing Liu, Ruijiao Liu, Yinghui Wang, Jinlei Chen, Yajie Yang, and Douli Ma. Image denoising searching similar blocks along edge directions. *Signal Processing: Image Communication*, 57:33–45, 2017.
- [23] Shaoping Xu, Xiaohui Yang, and Shunliang Jiang. A fast nonlocally centralized sparse representation algorithm for image denoising. *Signal Processing*, 131:99–112, 2017.
- [24] Viren Jain and Sebastian Seung. Natural image denoising with convolutional networks. In *Proc. Int. Conf. Neural Information Processing Systems*, pages 769–776, Vancouver, BC, 2009.
- [25] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 2392–2399, Providence, Rhode Island, 2012.
- [26] Forest Agostinelli, Michael R Anderson, and Honglak Lee. Adaptive multi-column deep neural networks with application to robust image denoising. In *Proc. Int. Conf. Neural Information Processing Systems*, pages 1493–1501, Lake Tahoe, NV, 2013.

- 
- [27] Yi-Qing Wang and Jean-Michel Morel. Can a single image denoising neural network handle all levels of Gaussian noise? *IEEE Signal Processing Letters*, 21(9):1150–1153, 2014.
- [28] D. R. K. Brownrigg. The weighted median filter. *Communications of the ACM*, 27(8):807–818, 1984.
- [29] Sung-Jea Ko and Yong Hoon Lee. Center weighted median filters and their applications to image enhancement. *IEEE Trans. Circuits and Systems*, 38(9):984–993, 1991.
- [30] Tong Sun and Yrjö Neuvo. Detail-preserving median based filters in image processing. *Pattern Recognition Letters*, 15(4):341–347, 1994.
- [31] Humor Hwang and Richard A Haddad. Adaptive median filters: New algorithms and results. *IEEE Trans. Image Processing*, 4(4):499–502, 1995.
- [32] Tao Chen and Hong Ren Wu. Adaptive impulse detection using center-weighted median filters. *IEEE Signal Processing Letters*, 8(1):1–3, 2001.
- [33] Shaomin Peng and Lori Lucke. Fuzzy filtering for mixed noise removal during image processing. In *Proc. IEEE Int. Conf. Fuzzy Systems*, pages 89–93, Orlando, FL, 1994.
- [34] Roman Garnett, Timothy Huegerich, Charles Chui, and Wenjie He. A universal noise removal algorithm with an impulse detector. *IEEE Trans. Image Processing*, 14(11):1747–1754, 2005.
- [35] Jian-Feng Cai, Raymond H Chan, and Mila Nikolova. Two-phase approach for deblurring images corrupted by impulse plus Gaussian noise. *Inverse Problems and Imaging*, 2(2):187–204, 2008.
- [36] Bin Dong, Hui Ji, Jia Li, Zuowei Shen, and Yuhong Xu. Wavelet frame based blind image inpainting. *Applied and Computational Harmonic Analysis*, 32(2):268–279, 2012.
- [37] Jun Liu, Xue-Cheng Tai, Haiyang Huang, and Zhongdan Huan. A weighted dictionary learning model for denoising images corrupted by mixed noise. *IEEE Trans. Image Processing*, 22(3):1108–1120, 2013.

- 
- [38] Jian Zhang, Debin Zhao, Ruiqin Xiong, Siwei Ma, and Wen Gao. Image restoration using joint statistical modeling in a space-transform domain. *IEEE Trans. Circuits and Systems for Video Technology*, 24(6):915–928, 2014.
- [39] Changzhong Zou and Youshen Xia. Bayesian dictionary learning for hyperspectral image super resolution in mixed Poisson–Gaussian noise. *Signal Processing: Image Communication*, 60:29–41, 2018.
- [40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. Int. Conf. Neural Information Processing Systems*, pages 1097–1105, Lake Tahoe, NV, 2012.
- [41] Jian Li, Jianjiang Feng, and C-C Jay Kuo. Deep convolutional neural network for latent fingerprint enhancement. *Signal Processing: Image Communication*, 60:52–63, 2018.
- [42] C. Dong, C. C. Loy, K. He, and X Tang. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.
- [43] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Proc. Int. Conf. Neural Information Processing Systems*, pages 1790–1798, Montreal, QC, 2014.
- [44] Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, and Hiroshi Ishikawa. Learning to simplify: fully convolutional networks for rough sketch cleanup. *ACM Transactions on Graphics (TOG)*, 35(4):121, 2016.
- [45] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.
- [46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

- 
- [47] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proc. Advances in Neural Information Processing Systems*, pages 3320–3328, Montral, Canada, 2014.
- [48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, Nevada, 2016.
- [49] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.
- [50] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
- [51] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. 32nd Int. Conf. Machine Learning*, pages 448–456, Lille, France, 2015.
- [52] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proc. Int. Conf. Artificial Intelligence and Statistics*, pages 315–323, Ft. Lauderdale, FL, 2011.
- [53] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proc. Int. Conf. Machine Learning*, pages 111–118, Haifa, Israel, 2010.
- [54] K Lang and M Witbrock. Learning to tell two spirals apart. In *Proc. 1988 Connectionists Models Summer School*, pages 52–59, San Mateo, CA, 1989.
- [55] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 4700–4708, Honolulu, Hawaii, 2017.

- 
- [56] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 2818–2826, Las Vegas, Nevada, 2016.
- [57] J Weston and C Watkins. Support vector machines for multi-class pattern recognition. In *Proc. European Symp. Artificial Neural Networks*, pages 219–224, Bruges, Belgium, 1999.
- [58] Zhou Wang, Alan Conrad Bovik, Hamid Rahim Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, 2004.
- [59] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. Int. Conf. Learning Representations*, pages 1–15, San Diego, CA, 2015.
- [60] Ronald W Schafer and Lawrence R Rabiner. A digital signal processing approach to interpolation. *Proceedings of the IEEE*, 61(6):692–702, 1973.
- [61] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 1646–1654, Las Vegas, Nevada, 2016.
- [62] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 1125–1134, Las Vegas, Nevada, 2016.
- [63] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Trans. Computational Imaging*, 3(1):47–57, 2017.
- [64] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva. Places2: A large-scale database for scene understanding. *ArXiv preprint arXiv:1610.02055*, 2016.
- [65] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg,

- and Li Fei-Fei. ImageNet large scale visual recognition challenge. *Int. J. Computer Vision*, 115(3):211–252, 2015.
- [66] Cnn codes and dataset. Released on November 29, 2017. URL: <https://github.com/tariqul-islam/Mixed-Noise-CNN>.
- [67] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proc. IEEE Int. Conf. Computer Vision*, pages 1026–1034, Las Condes, Chile, 2015.
- [68] Xavier Font-Aragones, Marcos Faundez-Zanuy, and Jiri Mekyska. Thermal hand image segmentation for biometric recognition. *IEEE Aerospace and Electronic Systems Magazine*, 28(6):4–14, 2013.
- [69] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proc. Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.
- [70] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proc. of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017.