EPFL

Thèse n° 8057

# Deep Image Restoration: Between Data Fidelity and Learned Priors

Présentée le 26 mars 2021

Faculté informatique et communications
Laboratoire d'images et représentation visuelle
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

**Majed EL HELOU**

Acceptée sur proposition du jury

Prof. P. Fua, président du jury
Prof. S. Süsstrunk, directrice de thèse
Prof. K. Egiazarian, rapporteur
Prof. A. Ortega, rapporteur
Prof. M. Unser, rapporteur

École polytechnique fédérale de Lausanne

2021

"Perception must, it seems, be a matter of seeing the present with stored objects from the past."
*— Richard L. Gregory, The Intelligent Eye*

"The perceiver may see the world before [they know] it."
*— Ralph Haber quoted by Vera J. Steiner, Notebooks of the Mind*

Dedicated to all who have, directly or indirectly, contributed to this thesis,
and to all the curious minds who might take a peek.

# Acknowledgements

Nothing worthwhile, let alone a doctoral thesis, is a single-person feat. Acknowledging all who made positive contributions towards this work is neither a tractable enumeration nor an accurately achievable objective, as words fail to convey the appropriate sense of appreciation. In pursuit of correctness, I sincerely thank absolutely all those who contributed to this thesis and dedicate it, in part, to them. Whether the contributions were directly from colleagues who actively collaborated on research, or from others with indirect, or even intractably indirect, contributions, they are all greatly appreciated and, hopefully, this was conveyed throughout the years.

All credits, and a big thank you, to Klara Fuss, Laura Jau, and Frederike Dümbgen for the German version of the thesis abstract.

Majed El Helou

# Abstract

Image restoration reconstructs, as faithfully as possible, an original image from a potentially degraded version of it. Image degradations can be of various types, for instance haze, unwanted reflections, optical or spectral aberrations, or other physically induced artifacts. Among the most fundamental restoration tasks is additive denoising, but also image inpainting and super-resolution. Denoising recovers an original image from an observed version containing a noise component over the image signal. It has significant theoretical importance as various problems can be reduced into a denoising problem, or reformulated to use a denoising solution. It also has significant practical importance due to its widespread use in imaging pipelines. Inpainting recovers image areas that are completely lost. Super-resolution increases the resolution of an image; in other words, it reconstructs an image with an effectively higher sampling rate and a larger-cutoff acquisition low-pass filter. To this end, it requires both effective deblurring and interpolation operations when viewing the problem from a spatial perspective.

The available methods for image restoration can be divided into two main categories; the classic restoration methods and the more recent deep neural network based approaches. With the advancement of deep learning, neural networks pushed the previous performance limits in image restoration, often at the expense of interpretability and reliability. Here, reliability means fidelity to the original image data. Classic image restoration is based in part on data fidelity and in part on priors that are manually designed, with the weighing between them also being manually chosen. Even though the distinction is often lost in the final output, the hallucinations induced by the prior are generally controllable and can be intuitively analyzed. This is, however, no longer the case with deep neural networks. These networks implicitly learn a prior and learn to be faithful to the original data, through the thousands or more of their hidden weights. Hence, control and interpretability over the contribution and nature of data fidelity and prior components are lost.

In this thesis, we analyze denoising and super-resolution networks in the frequency domain to gain further understanding over how image components and their inter-relations are learned and manipulated by the deep networks. Based on the obtained insights, a stochastic masking approach is presented to improve the learning. We also present a theoretical framework to evaluate a network's performance in learning the statistically optimal data fidelity and the optimal prior in a designed experimental setup. This framework is then generalized for denoising real image data by incorporating internal noise level estimation. Lastly, we present a framework that generalizes various families of classic restoration methods based on explicit optimizations and that can incorporate learned network priors. The framework also accounts for learning the fusion weights that balance between data fidelity and a learned prior, rather than a manually designed heuristic. As the framework enables us to disentangle these two components, the fusion weights are explicit,

and structurally given per pixel. These weights could benefit both the interpretability and the various downstream applications.

**Keywords:** image restoration, image denoising, learned priors, data fidelity, deep neural networks, convolutional neural networks, overfitting.

# Zusammenfassung

Bildrestaurierung rekonstruiert ein Originalbild aus einer potenziell degradierten Version so originalgetreu wie möglich. Bildverschlechterungen können unterschiedlicher Art sein, z. B. Dunst, unerwünschte Reflexionen, optische oder spektrale Aberrationen oder andere physikalisch bedingte Artefakte. Zu den grundlegendsten Restaurierungsaufgaben gehören additive Entrauschung, aber auch Inpainting und Super-Auflösung. Entrauschung rekonstruiert ein Originalbild aus einer beobachteten Version, welche eine Rauschkomponente über dem Bildsignal enthält. Da verschiedene Probleme auf ein Entrauschungs-Problem reduziert werden können, oder in abgeänderter Form durch Entrauschung gelöst werden können, ist diese von signifikanter theoretischer Bedeutung. Sie hat auch eine grosse praktische Relevanz, da sie in Bildverarbeitungspipelines weit verbreitet ist. Inpainting stellt Bildbereiche wieder her, die komplett verloren sind. Super-Resolution erhöht die Auflösung eines Bildes; mit anderen Worten rekonstruiert sie ein Bild mit einer effektiv höheren Abtastrate und einem Aufnahme-Tiefpassfilter mit grösserem Cutoff. Betrachtet man das Problem aus einer räumlichen Perspektive, sind zu diesem Zweck sowohl effektive Entschärfungs- als auch Interpolationsoperationen erforderlich.

Die verfügbaren Methoden zur Bildrestaurierung können in zwei Hauptkategorien unterteilt werden: die klassischen Restaurierungsmethoden und die neueren, auf tiefen neuronalen Netzwerken basierenden Ansätze. Mit dem Fortschritt von Deep Learning haben neuronale Netzwerke die bisherigen Leistungsgrenzen bei der Bildrestauration verschoben, oft auf Kosten von Interpretierbarkeit und Zuverlässigkeit. Zuverlässigkeit bezeichnet hier die Treue zu den ursprünglichen Bilddaten. Die klassische Bildrestauration basiert einerseits auf Datentreue und andererseits auf Priors, die manuell entworfen werden. Die Abwägung zwischen ihnen wird ebenfalls manuell gewählt. Auch wenn die Unterscheidung im Resultat oft verloren geht, sind die durch den Prior induzierten Halluzinationen im Allgemeinen kontrollierbar und können intuitiv analysiert werden. Dies ist bei tiefen neuronalen Netzwerken jedoch nicht mehr der Fall. Letztere lernen implizit, durch ihre unzähligen versteckten Gewichte, einen Prior und den Originaldaten treu zu bleiben. Die Kontrolle und Interpretierbarkeit über den Anteil und die Art der Datentreue und der Prior-Komponenten gehen deswegen verloren.

In dieser Doktorarbeit analysieren wir Entrauschungs- und Superresolutions-Netzwerke im Frequenzbereich, um ein besseres Verständnis darüber zu erlangen, wie Bildkomponenten und deren Zusammenhänge von tiefen Netzwerken gelernt und manipuliert werden können. Basierend auf den gewonnenen Erkenntnissen wird ein stochastischer Maskierungsansatz vorgestellt, welcher das Lernen verbessert. Des weiteren stellen wir einen theoretischen Rahmen vor, um die Fähigkeit eines Netzwerks zu evaluieren, die statistisch optimale Datentreue und Priors in einem Testversuch zu erlernen. Anschliessend

verallgemeinern wir diesen Rahmen für die Entrauschung von realen Bilddaten, indem wir eine interne Rauschpegelschätzung einbeziehen. Schließlich stellen wir ein Framework vor, das verschiedenste Familien klassischer Restaurationsmethoden, welche auf expliziten Optimierungen basieren, verallgemeinert und welches gelernte Netzwerk-Priors einbeziehen kann. Das Framework beinhaltet auch das Lernen der Fusionsgewichte, das ein Gleichgewicht zwischen der Datentreue und dem gelernten Prior herstellen, anstatt eine manuell entworfene Heuristik zu verwenden. Da das Framework es erlaubt, diese beiden Komponenten zu entflechten, sind die Fusionsgewichte explizit und strukturell pixelweise gegeben. Diese Gewichte können sowohl der Interpretierbarkeit als auch nachgelagerten Anwendungen zugutekommen.

**Schlüsselwörter:** Bildrestauration, Bildentrauschung, gelernte Priors, Datentreue, tiefe neuronale Netzwerke, neuronale Faltungsnetzwerke, Überanpassung.

# Contents

# List of Figures

# List of Tables

# Abbreviations and Symbols

## List of abbreviations

| Abbreviation | Description |
|---|---|
| SR | Super-resolution |
| CNN | Convolutional neural network |
| SFM | Stochastic frequency masking |
| LR | Low resolution |
| HR | High resolution |
| PSF | Point spread function |
| PSD | Power spectral density |
| WGN | White Gaussian noise |
| SNR | Signal-to-noise ratio |
| DCT | Discrete cosine transform |
| JPEG | Joint photographic experts group |
| AWGN | Additive white Gaussian noise |
| PSNR | Peak signal-to-noise ratio |
| LFM | Low frequency masking |
| HFM | High frequency masking |
| MSE | Mean squared error |
| BUIFD | Blind universal image fusion denoiser |
| MAP | Maximum a posteriori |
| MMSE | Minimum mean squared error |
| ReLU | Rectified linear units |
| SSIM | Structural similarity |
| BIGPrior | Bayesian integration of a generative prior |
| AuC | Area under curve |
| GT | Ground-truth |
| GAN | Generative adversarial network |
| LPIPS | Learned perceptual image patch similarity |

# List of symbols

| Symbol | Description |
|---|---|
| $\omega$ | Frequency value |
| $i$, $j$, $k$ | Indexing variables |
| $T$ | Sampling interval or downsampling rate |
| $q$, $z$ | Time- or space-domain signals |
| $Q$, $Z$ | Fourier-domain transforms |
| $\alpha$, $\lambda$, $\gamma$, $a$, $b$ | Parameter variables |
| $F^{LP}$, $F^{HP}$ | Low- and high-pass filters |
| $\sigma$, $\sigma_\delta$, $\sigma_x$, $\sigma_n$ | Standard deviation values |
| $f$ | Spatial frequency |
| $I^{HR}$ | High-resolution image |
| $\circledast$ | Convolution operator |
| $P(\cdot)$, $P(\cdot\|\cdot)$ | Probability and conditional probability |
| $x$ | Noise-free original image |
| $y$ | Noisy image |
| $n_G$ | Gaussian noise component |
| $n_P$ | Poisson noise component |
| $n$ | Additive noise |
| $\sim \mathcal{P}$ | Follows a Poisson distribution |
| $\Delta$ | Indexing offset |
| $\mathbb{E}$ | Statistical expectation |
| $R$ | Autocorrelation function |
| $\delta(\cdot)$ | Dirac delta function |
| $S_x(\cdot)$ | PSD of signal $x$ |
| $J$, $K$, $M$, $N$ | Counting/dimension size variables |
| $\delta_{k1}$ | Kronecker delta |
| $r_I$, $r_O$, $r_\omega$, $r_M$, $r_C$ | Radius values away from the DC component in DCT |
| $\delta$, $\delta_I$, $\delta_O$ | Offsets sampled from a half-normal distribution |
| $\epsilon$ | Very small constant value |
| $\mathbb{R}_+$ | Real positive numbers |
| $\hat{x}$ | Estimator of $x$ |
| $\bar{x}$ | Mean value of $x$ |
| $S$ | Gaussian-setup signal-to-noise ratio |
| $f(\cdot)$, $g(\cdot)$, $g^{-1}(\cdot)$ | Functions |
| $\mathcal{D}^{\mathcal{T}}$ | Training dataset |
| $\|\| \cdot \|\|_2$ | $\ell_2$ norm |
| $\|\| \cdot \|\|_1$ | $\ell_1$ norm |

| Symbol | Description |
|---|---|
| $\mathcal{N}(\mu, \sigma^2)$ | Normal distribution with mean $\mu$ and standard deviation $\sigma$ |
| $\mathcal{L}$, $\mathcal{L}_f$, $\mathcal{L}_G$ | Loss functions used in network training |
| $\theta_P$, $\theta_N$, $\theta_1$, $\theta_2$ | Network parameter weights |
| $F(\cdot)$, $\hat{F}(\cdot)$ | Fusion function and its estimator |
| $\mathbf{C}$ | Multi-input concatenation |
| $\odot$ | Pixel-wise or element-wise multiplication |
| $\beta$ | Fusion weight |
| $f'$, $f''$ | Manipulation functions |
| $\psi_d$ | Data fidelity penalty term |
| $\psi_p$ | Prior information penalty term |
| $\mathcal{T}$ | Signal transformation |
| $f_j$ | First-order derivative filters |
| $\otimes$ | Filter application, e.g. convolution |
| $G$ | Generative network |
| $G_1$, $G_2$ | Sub-network parts |
| $l$ | Network layer index |
| $v$ | Vector coordinates, for a dictionary space |
| $D$ | Dictionary |
| $d(\cdot, \cdot)$ | Distance function |
| $z_1$, ..., $z_N$ | Latent codes |
| $\alpha_1$, ..., $\alpha_N$ | Adaptive channel weights |
| $x^{inv}$ | Generative inversion of $x$ |
| $\phi$ | Pixel-wise fusion map |
| $\mathrm{span}(\cdot)$ | Spanning set |
| $\rho$ | Loss-balancing scalar |

# Chapter 1

# Introduction

## 1.1 Overview

Our experience in the *physical* world is governed by *observations* that are either sensed or induced. Along with these observations, there exists a set of often implicit *models* or *rules*. Whether these rules can be reasoned from an absolute truth and trusted without an appeal to innate nature or a deity [32], or without the need for an oracle that is external to the physical world [97], or whether any rules can even be truly separated from empirical observation [61], can turn into philosophical dilemmas. Drawing the frontiers between observation on one side, and the associated reasoning and assumed rules on the other side, is indeed not an obvious task.

Nevertheless, we can assume that the worth of the union of the two can theoretically be lower bounded by the worth of the best between them. From a theoretical point of view, even a 'set of observations' (if we perceive rules from an empiricist's perspective), combined with a current observation, can only be as bad as the restriction to only the current observation. This can be reasonably assumed, as the former case could be reduced to only the current observation through omission, at least in theory. It is therefore reasonable to use the former scenario, where the many *previous observations* can be translated into a form of rules or data models.

The physical world is a *stochastic* space, at least relative to our current imperfect observation, modeling, and inference capabilities. We can also consider it to be a *noisy* space, at least in relation to our imperfect observation and sensing capabilities. Therefore, observations are not always perfectly reliable, and models, even if assumed to be perfectly accurate, remain non-deterministic and cannot guarantee the accuracy of all consequent deductions. To navigate through this system and to reach conclusions that are as accurate as possible, we constantly juggle between our current observations and the models we *previously* had in mind, hence the associated name: *priors*. For humans, such models are not even necessarily derived thoroughly nor without bias, are not always in connection with our conscious mind processes, do not always reflect empirical evidence accurately, and are not necessarily well updated through time [5]. The exact frontier and interplay between observation-based reasoning and deduction, and the prior-based counterpart remains an open question, indeed a psychological or neuro-scientific one.

This frontier is, however, at the heart of imaging and image restoration methods. Imaging and image restoration can hardly be dissociated. Imaging is concerned with observing or scanning something physical to obtain visual information about it, in the form of images in the general sense (different number of spatial, temporal, or spectral dimensions). This scanning can be carried out for instance with visible light as in the human visual system, with other electromagnetic radiation, or with electrons. However, this scanning is less than perfect. For this reason, imaging and image restoration go hand in hand, where the latter corrects the shortcomings of the former. It is in image restoration that the interplay between observation-based information, which we call *data fidelity*, and prior-based information, is key. We first discuss image restoration and then dive into the interplay of the two types of information.

The purpose of image restoration is to fill the shortcomings of imaging systems. These shortcomings can be caused by the physical world itself, haze, rain, or other "occlusions", or by the physical limitations of the scanning medium, for instance, light scattering, or reflections for visible light. They can also be caused by the capture system, the optics, or the measurement hardware. We formulate these shortcomings in imaging in the form of deterministic or stochastic degradation models over an ideal image. The task of image restoration is then to undo the effects of the degradation models, thus restoring the ideal image from a degraded observation of it. For the fundamental task of additive image denoising, the degradation model that takes an ideal image $y$ and turns it into the observed image $x$ is a simple addition of a noise component $n$. The degradation model is thus formulated as $y = x + n$, and the objective of the restoration task is to obtain $x$, or equivalently $n$, from $y$. Prior information can take the form of a statistical model that governs $n$, but also any statistics over any component of $x$. In this setup, the data-fidelity component enforces our estimate solution for $x$ to be to a certain degree related to $y$, and the prior component leads our estimate to be more in accordance with the assumptions we make over $x$ and $n$. When $n$ is statistically close to zero, it is intuitive that data fidelity is crucial. In fact, $y$ is itself already a strong candidate estimator of $x$. On the contrary, when $n$ is close to infinite, the observation provides close to zero information on $x$. In this case, our estimator should rely on its prior information (*prior* to having made the observation) to obtain its best guess for $x$. In between these two extremes, the interplay between the two components is less intuitive, and so is their optimal combination to obtain a final restoration estimate. The factors affecting this interplay are the quality of our observation $y$, which is directly related to the strength of the noise $n$ for image denoising, and the quality of our prior information. We discuss these concepts more thoroughly in the remainder of the thesis, but we note here that they extend to other restoration tasks and that they emerge in the various image restoration algorithms in the literature.

Classic restoration methods most often implement their priors through different heuristics and assumptions derived experimentally from data. These priors are based on assumptions such as the presence of multiple similar patches in an image, or certain statistical distributions observed over image gradients. These methods then enforce their priors either directly through optimizations or indirectly as an effect of their algorithmic design. In recent years, artificial neural networks achieved remarkable progress in tasks such as image restoration. They derive their prior rules internally by observing and training over a large amount of data and are then able to make inference upon a novel observation. The nature of these networks, however, makes it similarly complex to analyze their inner-workings and the way they learn and manage the observations, or their **data fidelity**, and the learned data models, or **learned priors**. In this thesis, we study and build upon the interplay between these two aforementioned key components in image restoration.

Related work for the different topics is presented in each chapter. In the following section, we discuss the literature that is common across all the chapters, specifically the related work on denoising. In Section 1.3, we present basic background on anti-aliasing filtering that is relevant to the first chapter of the thesis.

## 1.2 Related Work on Denoising

Denoising is among the most fundamental of image restoration tasks because it is very important from an application perspective and due to its importance from a theoretical point of view. We address image denoising in each of the three following chapters in this thesis and, for readability, we group the common related work in this section. A denoising method is described as being *blind* if it is applicable to unknown variable test noise levels. A model is described as being *universal* if it consists of a single network or a single module for addressing all test cases, which is in contrast with approaches that store a multitude of sub-modules out of which one is selected depending on the current test input. We list a set of denoisers with their corresponding characteristics in Table 1.1 and discuss them further in the following paragraphs. The list includes our proposed BUIFD and BIGPrior methods that are presented in Chapter 3 and Chapter 4, respectively.

Classic image denoisers, such as PURE-LET [86] (specifically aimed at Poisson-Gaussian denoising), KSVD [2], WNNM [56], BM3D [29], and EPLL [165] (designed for Gaussian denoising), have the limitation that the noise level needs to be known at test time, or at least estimated [50]. Recent learning-based denoisers outperform the classic ones on Gaussian denoising [6, 103, 150]. But they require knowledge of the noise level [152] or even train multiple models for different noise levels [79, 151], which means that multiple models need to be pre-trained and stored. For instance, the recent method [151] that generalizes to image restoration tasks is a non-universal non-blind denoiser, where 25 denoising networks are used for noise levels below 50, and even training parameters are chosen based on the noise level. Similarly, the work of Remez *et al.* [105], which reaches PSNR results on par with the state of the art, is another non-universal non-blind example. To use better priors, images are first classified into a set of classes, and every single class has its specific deep network. The method is also not blind and is trained per noise level. Zhang *et al.* [153] present a universal non-blind network for multiple super-resolution degradations by denoising, deblurring, and by super-resolving images. They report that, though a blind version is more practical, their blind approach fails to perform consistently well because it cannot generalize.

For a model to work under blind settings and adapt to any noise level, a common approach is to train the denoiser network while varying the training noise level [6, 103, 150]. Having to know the exact noise level is indeed a serious limitation in practice for denoisers, and having to know it ahead of time, before training, is even more limiting. It is also a limitation, for example, when denoising images with a spatially varying noise level [152]. Another approach to avoiding both limitations, as presented in Chapter 3, is to predict the noise level, internally per pixel.

Other recent methods, for real-image denoising such as microscopy imaging [158], learn image statistics without requiring ground-truth samples. This is practical because ground-truth data can be extremely difficult and costly to acquire in, for instance, medical applications. Noise2Noise [80] learns to denoise from pairs of noisy images. The noise is assumed to be zero in expectation and decorrelated from the signal.

| Denoiser | Blind | Universal | Learning | Deep network | Formulated prior | Decoupled terms |
|---|---|---|---|---|---|---|
| BM3D [29] | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ |
| KSVD [2] | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ |
| WNNM [56] | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ |
| EPLL [165] | ✗ | ✓ | ✓* | ✗ | ✓ | ✗ |
| DnCNN [150] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| IRCNN [151] | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ |
| UNLNet [79] | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ |
| RIDNet [6] | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| FFDNet [152] | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| BUIFD [45] | ✓ | ✓ | ✓ | ✓ | ✗† | ✗ |
| BIGPrior [44] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

Table 1.1 – Representative non-comprehensive list of various image denoising methods and their different characteristics. A denoiser is blind if it does not require noise-level information at test time. And it is universal if the same model or algorithm is applied irrespective of the input. A denoiser involves learning when it trains, a priori on data, to obtain a dictionary or network weights. Classic methods are in the upper half of the table and do not rely on deep networks. A denoiser has a formulated prior if the prior is explicitly optimized and known, rather than indirectly induced through the nature of the denoiser or implicitly learned by a network. The denoiser has decoupled terms when the information related to data fidelity and to the prior are explicitly given. * For the denoising version of the EPLL method that produces the best results, a Gaussian mixture model is learned from training data. † Although our architecture is designed to internally push the network towards a Gaussian pixel prior, it is not explicitly enforced.

Therefore, unless the network memorizes it, the noise would not be predicted by it, and hence would be removed [80, 131]. Noise2Self [9], which is a similar but more general version of Noise2Void [72], also assumes the noise to be decorrelated, conditioned on the signal. The network learns from single noisy images, by learning to predict an image subset from a separate subset, again with the assumption that the noise is zero in expectation. Although promising, these two methods do not yet reach the performance of Noise2Noise.

## 1.3 Technical Background on Anti-aliasing

For completeness, in this section, we present some basic background on the anti-aliasing filtering necessary before downsampling a signal, and we focus on Gaussian kernels. The downsampling operation carried out over a signal, as we mention in Chapter 2, can cause a generally irreversible mixing of the frequency components of that signal in the final output. This mixing is referred to as aliasing. Avoiding or reducing this aliasing requires pre-filtering with a low-pass filter before downsampling, in order to suppress the frequency components that would result in this undesirable mixing. We look into the frequency-domain low-pass filtering for the commonly used one-dimensional Gaussian kernel

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}, \tag{1.1}$$

where $\sigma$ is the standard deviation. The Fourier transform $G(f)$ of the Gaussian kernel is then given by

$$G(f) = e^{-\frac{f^2}{2/\sigma^2}}. \tag{1.2}$$

If we set $p \in [0, 1]$ as the maximal accepted attenuation threshold, in the frequency region that is to be filtered to limit the effect of aliasing, and if we let the filtered region be delimited by $[f_c, +\infty]$ for a low-pass filter with cutoff $f_c$, then under these conditions we need

$$\sigma > \frac{\sqrt{-2\ln(p)}}{f_c}, \tag{1.3}$$

for the filter to be within the acceptable attenuation margin. $\sigma$ being by convention positive, and $f_c$ being imposed by the downsampling rate and Nyquist's theorem. We note that, as the Gaussian kernel's spectrum is not band-limited, the kernel cannot be sampled with a sufficiently small period for faithful reconstruction. For discrete images, different approximate discrete versions of a Gaussian blur kernel can be obtained by direct sampling or local integration. However, the choice of $\sigma$ is still dictated by the downsampling rate in order to avoid, or significantly attenuate, the aliasing problem. The value of $\sigma$ needs to be large enough to avoid or significantly attenuate the effect of aliasing, and its value is proportional to the downsampling rate (both of which are inversely related to the cutoff frequency).

In our experiments, we construct the discrete Gaussian filters with different standard deviations through the direct sampling of the corresponding continuous Gaussian functions. Other low-pass kernels are also used in the bicubic downsampling or in obtaining the real low-resolution images. The low-pass filter in the physical imaging domain is the point spread function (PSF) of the imaging system. The PSF is related to the lens and the aperture but also to the captured wavelength and to the depth of the point source in the scene [40]. It therefore varies from image to image, even within the same capture system, and is often modeled by a Gaussian kernel.

The main points, which are directly relevant to Chapter 2, are summarized as follows.

- A more severe blur, translated for instance into a larger standard deviation of the Gaussian kernel, erases a larger part of the high-frequency components but also has some attenuation effect on the low frequencies.
- A blind super-resolution method should be able to restore different ranges of high-frequency bands, because the different unknown degradation kernels affect different frequency components.

## 1.4 Contributions

In Chapter 2, we analyze, through the frequency domain, the elements that are preserved by the restoration networks from the observed data and the elements that are hallucinated with the learned prior. We investigate the nature of this prior that we show is a frequency conditional reconstruction that reflects the training degradation model. We also present a regularization technique that improves the robustness of the network's learning. Compared with the literature, this chapter provides novel insights and interpretations on the

underlying learning of the analyzed deep restoration networks. More specifically, we show that given an image with a range of low-frequency components, the networks hallucinate high-frequency components either by overfitting the training degradation model (hence dismissing the input conditioning) or by doing proper conditioning on the low frequencies. The networks achieve this proper conditioning after training with another contribution of this chapter, namely, our stochastic frequency masking regularization. Our approach enables the networks to avoid overfitting, to learn to perform a more general restoration, and hence to outperform the state-of-the-art results on various image super-resolution and denoising tasks.

In Chapter 3, we analyze under our pre-defined theoretical denoising setup the optimality of deep neural network learning, and we look into its generalization strength. We show that, by guiding the network internally into learning the statistically optimal prior, the generalization strength improves. We also demonstrate that, although not necessarily accurate for the real world, our design can be applied to the real-image denoising problem. In this chapter, we add to the literature a novel analysis and insights about the optimality potential and generalization of deep denoising. More specifically, we show that a network is able to reach the statistically optimal performance for a known, although simple, image prior. And we also show that it fails to directly generalize to data beyond its training experience. We also contribute a method that, building on our previous theoretical formulation, proposes an explicit internal learning of the noise level to regularize the interplay between data fidelity and learned prior inside the network. Our novel architecture improves the denoising performance over state-of-the-art additive Gaussian denoisers.

Lastly in Chapter 4, we present a general framework where we decouple the contributions of the data fidelity and the (learned) prior terms. A neural network is used for extracting prior information, which is then combined with a data fidelity term that relates through a bijection to the observed data. The combination weight is also learned, hence enabling a doubly adaptive fusion of the terms. Indeed it is adaptive, per input, both to the quality of the observation and to that of the learned prior. The framework we present also forms a generalization of large families of classic methods, as we discuss in the thesis, and it structurally provides a pixel-wise map that reveals the contribution of the prior-based hallucination to the final result. Our framework also improves the restoration performance on different tasks and consistently outperforms that of the underlying generative network inversion restoration.

# Chapter 2

# Frequency-Domain Study of Super-Resolution and Denoising Networks

In this chapter, we look into the internal learning mechanism of deep restoration networks. Our objective is to gain a better understanding of the inner-workings of the networks, in terms of the treatment of observed data and of the learning and application of internal data priors. We also improve the robustness of that underlying learning mechanism.

Super-resolution and denoising are ill-posed yet fundamental image restoration tasks. Under blind settings, the degradation kernel and/or the noise level are unknown. This makes restoration even more challenging, notably for learning-based methods, as they tend to overfit to the degradation seen during training. It is, however, unclear how this overfitting, and generally the restoration learning, can be formulated.

We present an analysis, in the frequency domain, of degradation-kernel overfitting in super-resolution and introduce a conditional-learning perspective that extends to both super-resolution and denoising. Building on our formulation, we propose a stochastic frequency masking of images used in training to regularize the networks and to address the overfitting problem. Our technique improves state-of-the-art methods on blind super-resolution with different synthetic kernels, real super-resolution, blind Gaussian denoising, and real-image denoising.

---

Our code and models are made publicly available at https://github.com/majedelhelou/SFM
This work is published in the European Conference on Computer Vision (ECCV), 2020. [46]

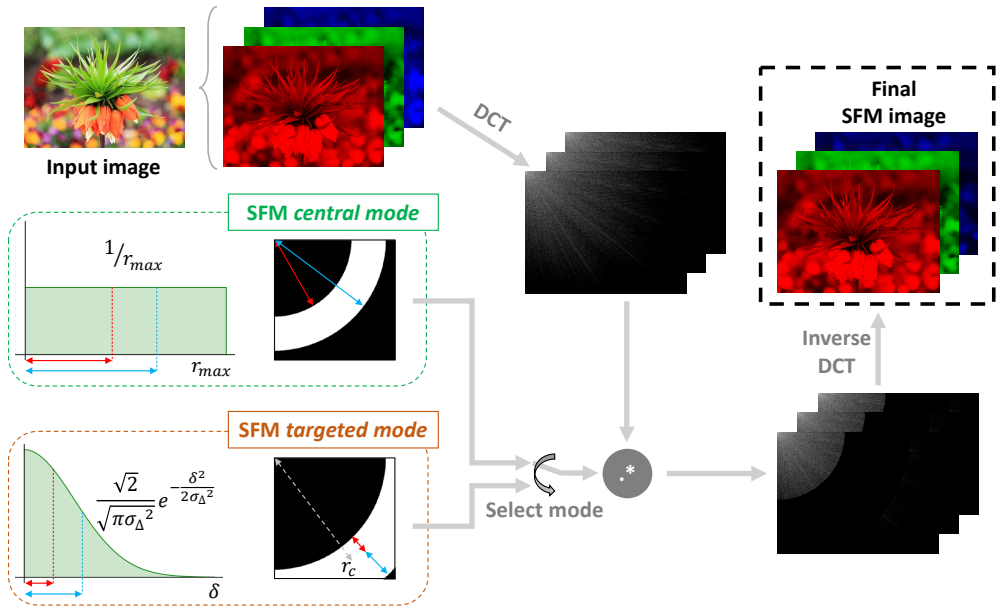



Figure 2.1 – Overview of Stochastic Frequency Masking (SFM). In the *central mode*, two radii values are sampled uniformly to delimit a masking area, and in the *targeted mode*, the sampled values delimit a quarter-annulus away from a target frequency. The obtained mask, shown with inverted color, is applied channel-wise to the discrete cosine transform of the image. We invert back to the spatial domain to obtain the SFM image that we use to train SR and denoising networks.

## 2.1 Introduction

Image super-resolution (SR) and denoising are fundamental restoration tasks widely applied in imaging pipelines. They are crucial in various applications, such as medical imaging [81, 100, 115], low-light imaging [21], astronomy [11], satellite imaging [12, 127], and face detection [57]. However, both are challenging ill-posed inverse problems. Recent learning methods, based on convolutional neural networks (CNN), achieve restoration performance better than classic approaches, both in SR and denoising. CNNs are trained on large datasets, sometimes real [164] but often synthetically generated with either one kernel or a limited set [134, 159]. They learn to predict the restored image or the residual between the restored target and the input [69, 150]. However, to be useful in practice, the networks should perform well on test images with unknown degradation kernels for SR, and unknown noise levels for denoising. Currently, they tend to overfit to the set of degradation models seen during training [39].

We investigate the SR degradation-kernel overfitting with an analysis carried out in the frequency domain. Our analysis reveals that an implicit conditional learning occurs in SR networks, specifically, the learning of residual high-frequency content given low frequencies. We additionally show that this result also extends to the denoising problem. Building on our insights, we present **S**tochastic **F**requency **M**asking (SFM): it stochastically masks frequency components of the *images used in training*. Our SFM method (Figure 2.1) is applied to a subset of the training images to regularize the network. It encourages the conditional learning to improve SR and denoising networks, notably when training under the challenging blind conditions. It can be applied during the training of *any* learning method, and has no additional cost at test time.

Our experimental results show that SFM improves the performance of state-of-the-art networks on blind SR and blind denoising. For SR, we conduct experiments on synthetic bicubic and Gaussian degradation kernels, and on real degraded images. For denoising, we conduct experiments on additive white Gaussian denoising and on real microscopy Poisson-Gaussian image denoising. SFM improves the performance of state-of-the-art networks on each of these tasks.

In this chapter, our contributions are summarized as follows. We present a frequency-domain analysis of the degradation-kernel overfitting of SR networks and highlight the implicit conditional learning that, as we also show, extends to denoising. We present SFM, a novel technique that regularizes the learning of SR and denoising networks by only filtering some training data. It enables the networks to better restore frequency components and to avoid overfitting. We empirically show that SFM improves the results of state-of-the-art learning methods on blind SR with different synthetic degradations, real-image SR, blind Gaussian denoising, and real-image denoising on high noise levels.

## 2.2 Related Work

**Super-resolution**. Depending on their image priors, SR algorithms can be divided into prediction models [113], edge-based models [20], gradient-profile pior methods [122] and example-based methods [51]. Deep example-based SR networks hold the state-of-the-art performance. Zhang *et al.* propose a very deep architecture based on residual channel attention in order to further improve these networks [159]. It is also possible to train in the wavelet domain to improve the memory and time efficiency of the networks [163]. Perceptual loss [64] and GANs [76, 134] are used to mitigate blur and to push the SR networks to produce more visually pleasing results. However, these networks are trained using a limited set of kernels, and studies have shown that they have poor generalization to unseen degradation kernels [54, 116]. To address blind SR, which is degradation-agnostic, recent methods propose to incorporate the degradation parameters, including the blur kernel, into the network [116, 151, 153, 154]. However, these methods rely on blur-kernel estimation algorithms hence have a limited ability to handle arbitrary blur kernels. The most recent methods, namely IKC [54] and KMSR [164], propose kernel estimation and modeling in their SR pipeline. However, it is hard to gather enough training kernels to cover the real-kernel manifold, while also ensuring effective learning and avoiding that these networks overfit to the chosen kernels. Recently, real-image datasets were proposed [19, 157] to enable SR networks to be trained and tested on high- and low-resolution (HR-LR) pairs that capture the same scene but at different focal lengths. These datasets are also limited to the degradations of only a few cameras and cannot guarantee that SR models trained on them would generalize to unseen degradations. Our SFM method, which builds on our degradation-kernel overfitting analysis and our conditional learning perspective, can be used to improve the performance of *all* the SR networks we evaluate, including those that estimate and model degradation kernels.

**Denoising**. We refer the reader to Section 1.2 for related work on the image denoising problem. We note that by regularizing the conditional learning defined from our frequency-domain perspective, our SFM method improves the high noise level results of *all* tested denoising networks, notably under blind settings.

One example that uses frequency bands in restoration is the method in [7]: it defines a prior based on a distance metric between a test image and a dataset of same-class images used for a deblurring optimization.

The distance metric computes differences between image frequency bands. In contrast, we apply frequency masking on training images in order to regularize deep restoration networks, and to improve performance and generalization. Spectral dropout [68] regularizes network activations by dropping out components in the frequency domain in order to remove the least relevant, whereas SFM regularizes training by promoting the conditional prediction of different frequency components through masking the training images themselves. The work most closely related to ours is a recent method proposed in the field of speech recognition [99]. The authors augment speech data in three ways, one of which is in the frequency domain. It is a random separation of frequency bands, which splits different speech components to enable the network to learn them one by one. A clear distinction with our approach is that we do not separate input components in order that they are each individually learned. Rather, we mask targeted frequencies from the *training* input to strengthen the conditional frequency learning, and we indirectly simulate the effect of a variety of kernels in SR and noise levels in denoising. The method we present is, to the best of our knowledge, the first frequency-based input *masking* method for regularizing SR and denoising training.

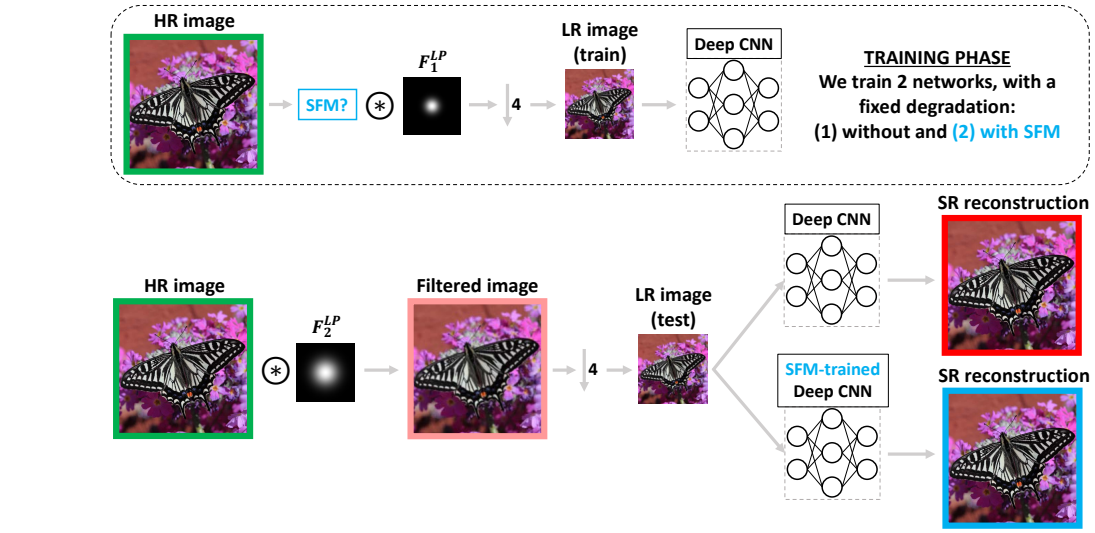
## 2.3 Frequency Perspective on SR and Denoising

### 2.3.1 Super-Resolution

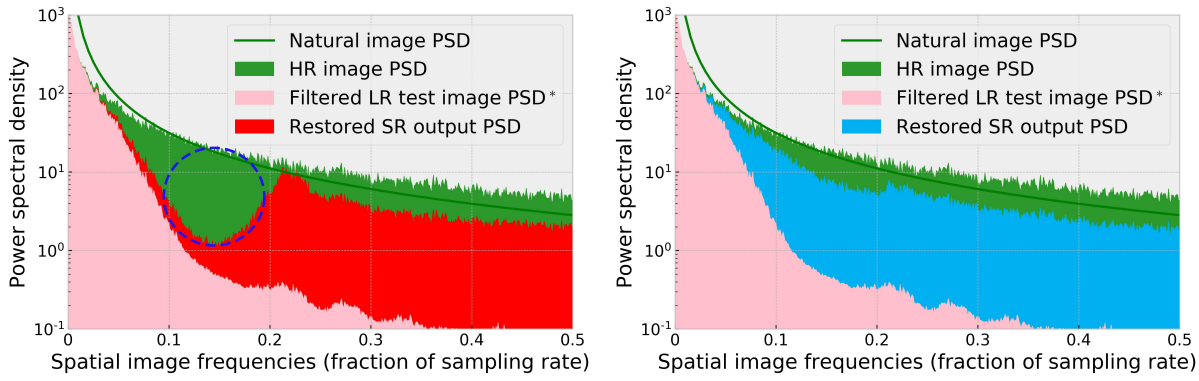
#### Preliminaries

Downsampling, a key element in modeling SR degradation, can be well explained in the frequency domain where it is represented by the sum of shifted and stretched versions of the frequency spectrum of a signal. Let $q$ be a one-dimensional discrete signal, e.g., a pixel row in an image, and let $z$ be a downsampled version of $q$ with a sampling interval $T$. In the discrete-time Fourier transform domain, with frequencies $\omega \in [-\pi, \pi]$, the relation between the transforms $Q$ and $Z$ of the signals $q$ and $z$, respectively, is given by $Z(\omega) = \frac{1}{T} \sum_{k=0}^{T-1} Q((\omega + 2\pi k)/T)$. The $T$ replicas of $Q$ can overlap in the high frequencies and cause aliasing. Aside from complicating the inverse problem of restoring $q$ from $z$, aliasing can create visual distortions. Therefore, before downsampling, low-pass filtering is applied to attenuate if not to completely remove the high-frequency components that would otherwise overlap.

These low-pass filtering blur kernels are applied through a spatial convolution over the image. The set of real kernels spans only a subspace of all mathematically possible kernels. This subspace is, however, not well-defined analytically and, in the literature, is often limited to the non-comprehensive subspace spanned by 2D Gaussian kernels. Thus, many SR methods model the anti-aliasing filter as a 2D Gaussian kernel, in an attempt to mimic the point spread function (PSF) of capturing devices [37, 114, 140]. In practice, even a single imaging device results in multiple kernels, depending on its settings [40]. For real images, the kernel can also be different from a Gaussian kernel [39, 54]. The essential point is that the anti-aliasing filter causes the loss of high-frequency components, and that this filter can differ from image to image.




(a) Experimental setup




(b) Without SFM

(c) With SFM

Figure 2.2 – (a) Overview of our experimental setup, with image border colors corresponding to the plot colors shown in (b,c). We train two versions of the same network on the same degradation kernel ($F_1^{LP}$ anti-aliasing filter), one without and one with SFM, and we test them using $F_2^{LP}$. (b) Average PSD (power spectral density) of HR images in green fill, with a green curve illustrating a typical natural-image PSD ($\alpha = 1.5$ [130]). The pink fill illustrates the average PSD of the low-pass filtered LR test images (**shown before downsampling for better visualization*). In red fill is the average PSD of the restored SR output image. The blue-dashed circle highlights the learning gap due to degradation-kernel overfitting. (c) The same as (b), except that the output is that of the network trained with SFM. The results are averaged over 100 random samples.

### Frequency Visualization of SR Reconstructions

SR networks tend to overfit the blur kernels used in the degradation for obtaining the training images [153]. To understand this phenomenon, we analyze in this section the relation between the frequency-domain effect of a blur kernel and the reconstruction of SR networks. We carry out the following experiment with a network trained with a unique and known blur kernel. We use the DIV2K [1] dataset to train a 20-block

RRDB [134] x4 SR network with images filtered by a Gaussian blur kernel called $F_1^{LP}$ (standard deviation $\sigma = 4.1$), shown in the top row of Figure 2.2(a). Then, to analyze the potential network overfitting, we run an inference on 100 test images that are filtered with a different Gaussian blur kernel called $F_2^{LP}$ ($\sigma = 7.4$), shown in the bottom row of Figure 2.2(a).

We present a frequency-domain visualization in Figure 2.2(b). The power spectral density (PSD) is the distribution of frequency content in an image. The typical PSD of an image (green curve) is modeled as $1/f^{\alpha}$, where $f$ is the spatial frequency, with $\alpha \in [1, 2]$ and varying depending on the scene (natural vs. man-made) [18, 49, 129, 130]. The $1/f^{\alpha}$ trend is visible in the PSD of HR images (green fill). The degraded LR test images are obtained with a low-pass filter on the HR image, before downsampling, and their frequency components are mostly low frequencies (pink fill). The SR network outputs contain high-frequency components restored by the network (red fill). However, these frequencies are mainly above $0.2\pi$, which is the range that was filtered out by the kernel used in creating the *training* LR images. The low-pass kernel used in creating the test LR images filters out a larger range of frequencies; it has a lower cutoff than the training kernel (the reverse case is also problematic and is illustrated in the following paragraph). This causes a gap of missing frequency components not obtained in the restored SR output; it is illustrated with a blue-dashed circle in Figure 2.2(b). The results suggest that an implicit conditional learning takes place in the SR network; we expand further on this in the following section. The results of the network trained with 50% SFM (masking applied to half of the training set) are shown in Figure 2.2(c). A key observation is that the missing frequency components are predicted to a far better extent when the network is trained with SFM.

We further vary the training degradation kernel $F_1^{LP}$ and the testing degradation kernel $F_2^{LP}$, and we repeat the same experiment. In Figure 2.3, we present our frequency visualization, with networks trained and tested using different degradation kernels, with and without 50% SFM. For the SR networks trained without our proposed SFM, the restored SR output images from the networks (red fill) have gaps of missing frequency components when the testing degradation kernel has a cutoff frequency lower than the training degradation kernel (larger Gaussian standard deviation). As explained in Section 2.3.1, when the testing degradation kernel actually has a cutoff larger than the training degradation kernel, the SR networks trained without SFM reconstruct redundant frequency components in the restored SR output (instead of a gap, we see a very clear surplus over the ground-truth PSD). This is shown in the plots below the diagonal of Figure 2.3. The missing and the redundant frequency components are largely resolved by the same network architecture trained with SFM (light blue fill).

### Implicit Conditional Learning

As we explain in the Preliminaries of Section 2.3.1, the high-frequency components of the original HR images are removed by the anti-aliasing filter. If this filter is *ideal*, it means that the low-frequency components are not affected and that the high frequencies are removed perfectly. We propose that the SR networks implicitly learn a conditional probability

$$P\left(I^{HR} \circledast F^{HP} \mid I^{HR} \circledast F^{LP}\right), \tag{2.1}$$

where $F^{HP}$ and $F^{LP}$ are ideal high-pass and low-pass filters, applied to the high-resolution image $I^{HR}$, and $\circledast$ is the convolution operator. The low- and high-frequency ranges are theoretically defined as $[0, \pi/T]$ and $[\pi/T, \pi]$, which is the minimum condition (largest possible cutoff) to avoid aliasing for a downsampling

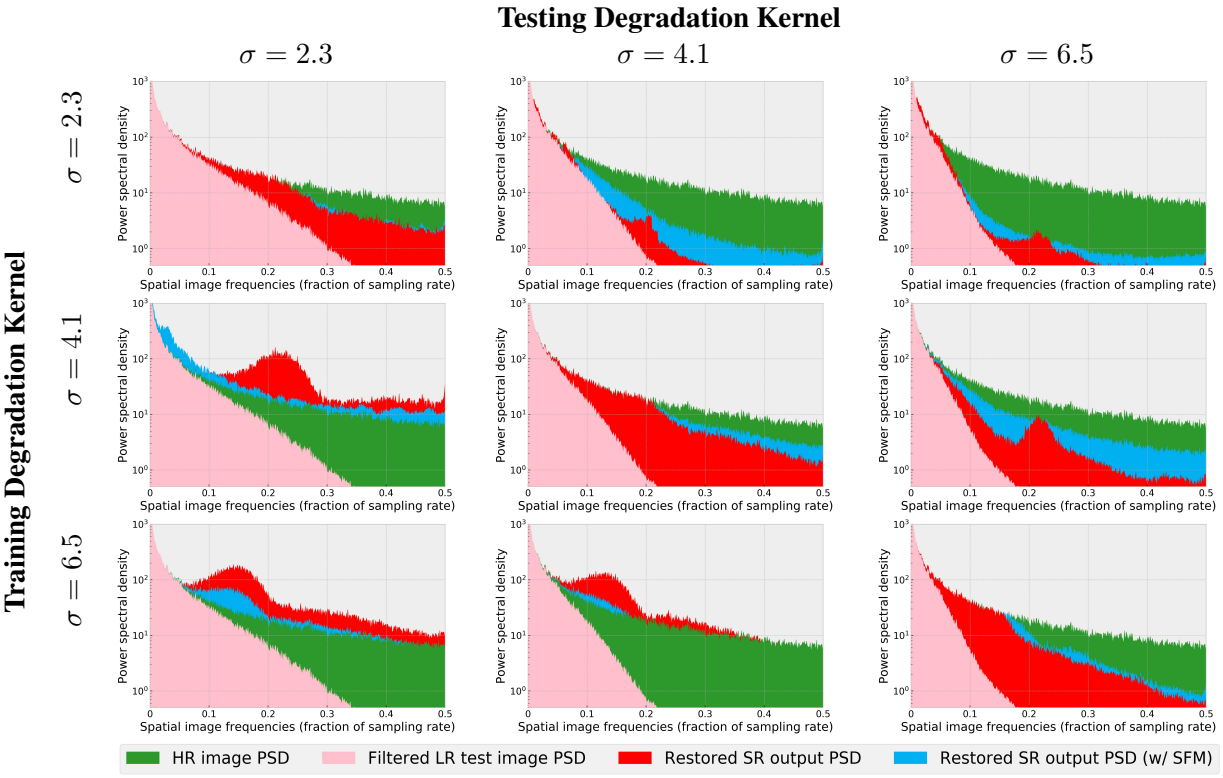



Figure 2.3 – Frequency visualization of SR reconstructions with different training and testing degradation kernels. We use the same experimental settings as in Section 2.3.1. We train a 20-block RRDB [134] x4 SR network with images filtered by different Gaussian blur kernels, one at a time; and we evaluate it with three different Gaussian blur kernels. Results are averaged over 100 random samples. We can see that SFM improves the SR reconstructions, as the PSD of the restored images is closer to the ground-truth than without SFM, and follows the average PSD power law [18, 49, 129, 130]. SFM largely resolves the gap of missing frequencies when the test kernel has a frequency cutoff lower than the training kernel (plots above the diagonal). SFM also resolves the issue of redundant frequency components restored when the test kernel has a frequency cutoff value larger than the training kernel (plots below the diagonal). Furthermore, we note that SFM slightly improves even the methods *trained and tested on the same degradation kernel* (the three plots on the diagonal).

rate $T$. The components of $I^{HR}$ that survive the low-pass filtering are the same frequencies contained in the LR image $I^{LR}$, when the filters $F$ are ideal. In other words, the frequency components of $I^{HR} \circledast F^{LP}$ are those remaining in the LR image that is the network input.

The anti-aliasing filters are, in practice, not ideal, which results in (a) some low-frequency components of $I^{HR}$ being attenuated, and (b) some high frequencies surviving the filtering and causing aliasing. Typically, the main issue is the first issue (a), because filters are chosen in a way that removes the visually disturbing aliasing at the expense of attenuating some low frequencies. We analyze this practical case with non-ideal anti-aliasing filters in the following: With the downsampling filters being non-ideal, a frequency-domain trade-off imposes itself. Some high frequencies are not completely removed and/or some low/mid frequencies are attenuated. In practice, aliasing is removed as it is very visually disturbing; and this is done at the cost of losing some information in the low-frequency range. We define $F_o^{LP}$ to be a practical non-ideal low-pass filter. The underlying conditional probability distribution, needed to recover the missing information, becomes

$$P\left(I^{HR} - I^{HR} \circledast F_o^{LP} \mid I^{HR} \circledast F_o^{LP}\right), \tag{2.2}$$

where the frequency components of $(I^{HR} \circledast F_o^{LP})$ are those remaining in the low-resolution input image. We note here the similarity with the residual learning introduced in [150]. The difference relative to the ideal-filter case is that $(1 - F_o^{LP})$ no longer corresponds to an ideal high-pass filter. Nonetheless, we can separate the frequency components of the residual image $I^{HR} - I^{HR} \circledast F_o^{LP} =$

$$(I^{HR} - I^{HR} \circledast F_o^{LP}) \circledast F^{LP} + (I^{HR} - I^{HR} \circledast F_o^{LP}) \circledast F^{HP}, \tag{2.3}$$

where again $F^{HP}$ and $F^{LP}$ are complementary *ideal* high-pass and low-pass filters. We note two properties of the filters, first,

$$F_o^{LP} \circledast F^{LP} = F_o^{LP}, \tag{2.4}$$

which is true for any anti-aliasing filter $F_o^{LP}$ that completely removes aliasing effects and for any ideal low-pass filter $F^{LP}$, and second,

$$F_o^{LP} \circledast F^{HP} = 0. \tag{2.5}$$

The proof of Equation (2.4) becomes straightforward when translated into the frequency domain, where the convolution becomes an element-wise product. Indeed, $F^{LP}$ is an ideal filter that does not affect low frequencies and completely removes high frequencies. Also, $F_o^{LP}$ removes aliasing hence removes all high frequencies (above $\pi/T$, for a downsampling rate $T$). Effectively, applying $F^{LP}$ on $F_o^{LP}$ only removes the high frequency values which are already zero. The proof of Equation (2.5) can be derived, using Equation (2.4) and the fact that filters are ideal, as follows

$$F_o^{LP} \circledast F^{HP} = F_o^{LP} \circledast (1 - F^{LP}) = F_o^{LP} - F_o^{LP} = 0. \tag{2.6}$$

By expanding Equation (2.3) and using Equation (2.4) and Equation (2.5), we can derive that $I^{HR} - I^{HR} \circledast F_o^{LP} =$

$$\underbrace{I^{HR} \circledast F^{LP} - I^{HR} \circledast F_0^{LP}}_{low-freq\ residual} + \underbrace{I^{HR} \circledast F^{HP}}_{high-freq}. \tag{2.7}$$

The interesting result is the separation between low-frequency components and high-frequency ones, as we

assume that they are conditionally independent (conditioned on $I^{HR} * F_o^{LP}$). With this assumption, we can factorize Equation (2.2) into the two factors

$$\begin{cases} P\left(I^{HR} \circledast F^{LP} - I^{HR} \circledast F_0^{LP} \mid I^{HR} \circledast F_o^{LP}\right) \\ P\left(I^{HR} \circledast F^{HP} \mid I^{HR} \circledast F_o^{LP}\right). \end{cases} \tag{2.8}$$

We first note that this also leads us to an implicit conditional distribution for predicting the high frequencies

$$P\left(I^{HR} \circledast F^{HP} \mid I^{HR} \circledast F_o^{LP}\right), \tag{2.9}$$

which is the same as Equation (2.1) except for the conditional term. Indeed, instead of learning to predict the high-frequency components given the low-frequency ones, the network is given a degraded version of the low frequencies. The network must learn to predict the residual of the degraded low frequencies

$$P\left(I^{HR} \circledast F^{LP} - I^{HR} \circledast F_0^{LP} \mid I^{HR} \circledast F_o^{LP}\right). \tag{2.10}$$

Although the target components predicted through the distribution in Equation (2.9) are the same, irrespective of the degradation kernel $F_o^{LP}$, the target residual predicted through the distribution in Equation (2.10) depends on this kernel. Hence, the network trained using this degradation kernel could overfit and always produce the same residual, irrespective of the degradation of the test image. This issue is illustrated in Figure 2.3. In Figure 2.3, the networks that are tested on images degraded with a kernel that removes more frequencies than the training kernel do not predict the missing frequency components (plots above the diagonal). Inversely, the networks tested on images degraded with a kernel that removes fewer frequency components end up adding residual frequency components that are already in the input image (plots below the diagonal). The former can be visualized as gaps of missing frequencies, and the latter can be seen as an addition of redundant frequency components (Figure 2.3).

Therefore, even with non-ideal filters, there is still conditional and residual learning components to predict a set of high-frequencies. These frequencies are, however, conditioned on a set of low-frequency components potentially attenuated by the non-ideal filter we call $F_o^{LP}$. This filter fully removes aliasing artifacts but can affect the low frequencies. Hence, the distribution can be defined by the components

$$P\left(I^{HR} \circledast F^{HP} \mid I^{HR} \circledast F_o^{LP}\right), \quad P\left(I^{HR} \circledast F^{LP} - I^{HR} \circledast F_0^{LP} \mid I^{HR} \circledast F_o^{LP}\right). \tag{2.11}$$

This can again be observed through our results in Figure 2.2. The SR network trained with degradation kernel $F_1^{LP}$ ($\sigma = 4.1$ in our experiment) restores the missing high frequencies of $I^{HR}$ that would be erased by $F_1^{LP}$. However, this is the case even though the test image is degraded by $F_2^{LP} \neq F_1^{LP}$. As $F_2^{LP}$ ($\sigma = 7.4$) removes a range of frequencies wider than $F_1^{LP}$, not predicted by the network, these frequencies remain missing. We observe a gap in the PSD of the output, highlighted by a blue-dashed circle. This illustrates the degradation-kernel overfitting issue from a frequency-domain perspective. We also note that these missing frequency components are restored by the network trained with SFM.

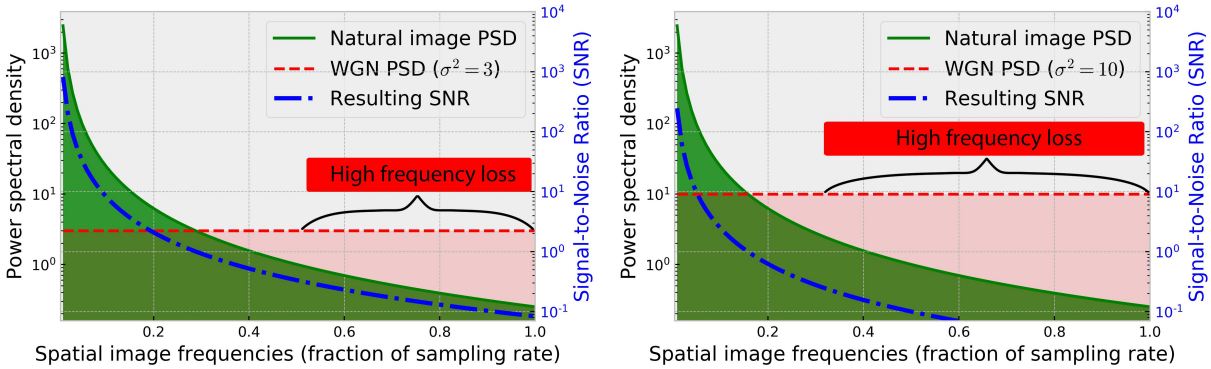



Figure 2.4 – Natural image PSD follows a power law as a function of spatial frequency. The plotted examples follow a power law with $\alpha = 2$ [130] and additive WGN ($\sigma^2 = 3$ on the left, and $\sigma^2 = 10$ on the right). The resulting SNR in the noisy image is exponentially smaller the higher the frequency, effectively causing a high frequency loss. The higher the noise level, the more frequency loss is incurred, and the more similar denoising becomes to our SR formulation.

### 2.3.2 Extension to Denoising

We highlight a connection between our conditional learning proposition and denoising. As discussed in Section 2.3.1, the average PSD of an image can be approximated by $1/f^{\alpha}$. The Gaussian noise samples added across pixels are independent and identically distributed. The PSD of the additive white Gaussian noise is uniform. Figure 2.4 shows the PSD of a natural image following a power law with $\alpha = 2$, that of white Gaussian noise (WGN), and the resulting signal-to-noise ratio (SNR) when the WGN is added to the image. The resulting SNR decreases proportionally to $1/f^{\alpha}$.

The relation between SNR and frequency shows that with increasing frequency, the SNR becomes exponentially small. In other words, high frequencies are almost completely overtaken by the noise, whereas low frequencies are much less affected by it. And, *the higher the noise level, the lower the starting frequency beyond which the SNR is significantly small*, as illustrated by Figure 2.4. This draws a direct connection to our SR analysis. Indeed, in both applications there exists an implicit conditional learning to predict lost high-frequency components, given low-frequency ones that are less affected.

So far, we have analyzed the power spectral density functions of white Gaussian noise and of natural images to draw a parallel between denoising and our super-resolution formulation. Now, we discuss the PSD in the case of additive Gaussian noise and Poisson noise. In the presence of Poisson-Gaussian noise, i.e., both additive Gaussian and Poisson noise components, the measured pixel intensity $y$ at pixel $i$ is given by

$$y[i] = x[i] + n_P(x[i]) + n_G, \tag{2.12}$$

where $x[i]$ is the noise-free signal at pixel $i$, $n_G$ is a noise sample taken from a Gaussian distribution with standard deviation tied to the Gaussian noise level, and $n_P(\lambda)$ is a noise sample from a Poisson distribution with mean $\lambda$ *from which we subtract* $\lambda$. This means that $(x[i] + n_P(x[i])) \sim \mathcal{P}(a \cdot x[i])$ [50], for some $a > 0$.

The additive Gaussian noise samples are modeled as independent and identically distributed with zero

mean [137]. Hence, the corresponding PSD is uniform and depends only on the noise level (the samples are taken from a Gaussian stochastic process of length equal to the number of pixels, they are uncorrelated and have zero mean). The PSD can be calculated from the co-variance function and is equal to $\sigma^2$ where $\sigma$ is the standard deviation of the Gaussian distribution. The additive Poisson noise is, however, not necessarily white with a uniform PSD. The different Poisson noise samples are taken from different probability distributions. Together with the clean signal, $(x[i] + n_P(x[i]))$, they are taken from a Poisson distribution with mean $a \cdot x[i]$ that varies with $i$. The Poisson noise components, signal aside, have zero mean. The auto-correlation function for the noise (referred to as $n$ in what follows) is

$$R_n(\Delta) = \mathbb{E}[n[i]n[i+\Delta]] = \begin{cases} \mathbb{E}[n[i]^2], \Delta = 0 \\ \mathbb{E}[(y[i] - x[i])(y[i+\Delta] - x[i+\Delta])], o.w., \end{cases} \tag{2.13}$$

and we condition then run expectation on $x$ for both terms (entire $x$ vector). The first terms leads to the variance of $n[i]$ conditioned on $x$, whose conditional distribution is a Poisson distribution of mean $ax[i]$ but that is zero-shifted. After running the expectation over $x$ we have $a\mathbb{E}[x[i]] = a\mathbb{E}[x]$. The second term then becomes

$$\begin{gathered} \mathbb{E}[(y[i] - x[i])(y[i+\Delta] - x[i+\Delta])] = \mathbb{E}_x[x[i]x[i+\Delta]] + ... \\ \mathbb{E}_x[-x[i]\mathbb{E}[y[i+\Delta]|x] - x[i+\Delta]\mathbb{E}[y[i]|x] + \mathbb{E}[y[i]y[i+\Delta]|x]], \end{gathered} \tag{2.14}$$

which, because $y$ instances are independent conditioned on $x$, and because $\mathbb{E}[y[i]|x] = ax[i]$, leads to

$$R_n(\Delta) = a\mathbb{E}[x]\delta(\Delta) + [R_x(\Delta) - aR_x(\Delta) - aR_x(\Delta) + a^2R_x(\Delta)](1 - \delta(\Delta)), \tag{2.15}$$

where $\delta(\cdot)$ is the Dirac delta function. Taking the Fourier transform on both sides, we obtain that the SNR is given by

$$\frac{S_x(f)}{a\mathbb{E}[x] + (1-a)^2 S_x(f) - (1-a)^2\mathbb{E}[x^2]}, \tag{2.16}$$

where $f$ is the frequency and $S_x(f)$ is the PSD of $x$. This shows that, although to a lesser degree than with purely Gaussian noise, the SNR goes to zero at higher frequencies as the PSD of $x$ itself goes to zero. And we lastly also note that in the case of strong Poisson noise, the Poisson component can itself be well approximated by a Gaussian [50], with the error decreasing with the increasing strength of the Poisson component.

## 2.4 Stochastic Frequency Masking (SFM)

### 2.4.1 Motivation and Implementation

The purpose of SFM is to improve, whether for SR or denoising, the networks' prediction of high frequencies, given lower ones. We achieve this by stochastically masking high-frequency bands from some of the training images in the learning phase, in order to encourage the conditional learning of the network. Our masking is carried out by transforming an image to the frequency domain using the Discrete Cosine Transform (DCT) type II [3, 120], by multiplying channel-wise with our stochastic mask, and lastly by transforming the image back (Figure 2.1). Specifically, we use the discrete cosine transform DCT type 2, also called DCT-II. The

DCT-II of a one-dimensional discrete signal $q$ of length $N$ is defined by

$$z[k] = \sqrt{\frac{2}{N}} \sum_{j=1}^{N} q[j] \frac{1}{\sqrt{1+\delta_{k1}}} \cos\left(\frac{\pi}{2N}(2j-1)(k-1)\right), \tag{2.17}$$

where $\delta_{k1}$ is the Kronecker delta [3, 120]. The inverse is obtained by swapping $j$ and $k$, as the DCT is orthogonal. The two-dimensional DCT is obtained by applying the DCT along the first dimension then along the second, and it forms the basis of the JPEG compression standard [132]. The DCT-II of a length-$N$ discrete sequence is equivalent to the DFT of a sequence of length $2N$, created by mirroring the original length-$N$ sequence to avoid DFT artifacts [87]. Such artifacts are due to the fact that the signal is assumed to be circulantly continuous by the DFT. We mediate this issue by using the DCT-II that we adopt in our proposed method.

We define frequency bands in the DCT domain over quarter-annulus areas, in order to cluster together similar-magnitude frequency content. Therefore, the SFM mask is delimited with a quarter-annulus area by setting the values of its inner and outer radii. We define two masking modes, the *central mode* and the *targeted mode*.

In the *central mode*, the inner and outer radius limits $r_I$ and $r_O$ of the quarter-annulus are selected uniformly at random from $[0, r_M]$, where $r_M = \sqrt{a^2+b^2}$ is the maximum radius, with $(a, b)$ being the dimensions of the image. We ensure that $r_I < r_O$ by permuting the values if $r_I > r_O$. With this mode, the resulting probability of a given frequency band $r_\omega$ to be masked is

$$P(r_\omega = 0) = P(r_I < r_\omega < r_O) = 2\left(\frac{r_\omega}{r_M} - \left(\frac{r_\omega}{r_M}\right)^2\right), \tag{2.18}$$

meaning the central bands are the more likely ones to be masked, with the likelihood *slowly* decreasing for higher- or lower-frequency bands. In the *targeted mode*, a target frequency $r_C$ is selected, with a parameter $\sigma_\delta$. The quarter-annulus is delimited by $[r_C - \delta_I, r_C + \delta_O]$, where $\delta_I$ and $\delta_O$ are independently sampled from the half-normal distribution $f(\delta) = \sqrt{2}/\sqrt{\pi\sigma_\delta^2} e^{-\delta^2/(2\sigma_\delta^2)}, \forall \delta \geq 0$. Therefore, with this mode, the frequency $r_C$ is always masked, and the frequencies away from $r_C$ are increasingly less likely to be masked, with a normal distribution decay.

We use the *central mode* for SR networks and the *targeted mode* with a high target $r_C$ for denoisers (Figure 2.1). The former has a slow probability decay that covers wider bands, whereas the latter has an exponential decay adapted for targeting specific narrow bands. In both settings, the highest frequencies are most likely to be masked. The *central mode* masks the highest frequencies in SR, because central frequencies are the highest ones remaining after the anti-aliasing filter is applied. It is also worth noting that SFM simulates the effect of different blur kernels by stochastically masking different frequency bands. Blur kernels are typically defined spatially through convolution. They are defined inside $\mathbb{R}^{K\times K}$ for kernels with support $K$. Synthesizing all possible kernels in this space is computationally impractical. It is also not sensible because realistic kernels form only a sub-space of $\mathbb{R}^{K\times K}$ that is, however, not well-defined analytically. Translating a blur kernel to the frequency domain, for instance with the DCT, provides a dissected view of the kernel's effect. A kernel acts in a multiplicative manner over every frequency band. Therefore, the effect of applying a certain blur kernel can be distributed into a basis of independent multiplications on

(a) Input ($\sigma = 2.9$) (b) RCAN [159] (c) ESRGAN [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) ESRGAN+SFM (h) IKC + SFM

Figure 2.5 – Cropped SR results (x4) of different methods (top row), and with our SFM added (bottom row), for image 0844 of DIV2K. The visual quality improves for all methods when trained with SFM (images best viewed on screen).

every frequency component. If we segment frequency components into a set of $M$ bands, then $\forall \epsilon \in \mathbb{R}_+, \exists M$ large enough such that the absolute error of approximating the kernel function with a set of constant steps of equal width is $< \epsilon$ (limit of a Riemann sum to a Riemann integral). Hence, we can simulate the effect of different kernels, through a finite set of $M$ steps in the frequency domain, with a controllable trade-off between accuracy and computation. We approximate this filtering effect in a binary way (our multiplicative step values are 0 or 1), with our SFM, by stochastically masking different frequency bands. Therefore, SFM uses a spanning set for the space of degradation kernels, which improves generalization to unknown kernels.

### 2.4.2 Learning SR and Denoising with SFM

We apply SFM *only* on the input training data. For the simulated-degradation data, SFM is applied in the process of generating the LR inputs. We apply SFM on HR images, before applying the degradation model to generate the LR inputs (blur kernel and downsampling). The target output of the network remains the original HR images. For real images where the LR inputs are given and the degradation model is unknown, we apply SFM on the LR inputs and keep the original HR images as ground-truth targets. Therefore, the networks trained with SFM do not use any additional data relative to the baselines. We apply *the same* SFM settings for all deep learning experiments. During training, we apply SFM on 50% of the training images, using the *central mode* of SFM, as presented in Section 2.4.1. Ablation studies with other rates are in our *Supplementary Material*. We add SFM to the training of the original methods, with no other modification.

| | Test blur kernel ($g_\sigma$ is a Gaussian kernel, standard deviation $\sigma$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bicubic | $g_{1.7}$ | $g_{2.3}$ | $g_{2.9}$ | $g_{3.5}$ | $g_{4.1}$ | $g_{4.7}$ | $g_{5.3}$ | $g_{5.9}$ | $g_{6.5}$ |
| RCAN [159] | 29.18 | 23.80 | 24.08 | 23.76 | 23.35 | 22.98 | 22.38 | 22.16 | 21.86 | 21.72 |
| RCAN+SFM | **29.32** | **24.21** | **24.64** | **24.19** | **23.72** | **23.27** | **22.54** | **22.23** | **21.91** | **21.79** |
| IKC [54] | **27.81** | 26.07 | 26.15 | 25.48 | 25.03 | 24.41 | 23.39 | 22.78 | 22.41 | 22.08 |
| IKC+SFM | 27.78 | **26.09** | **26.18** | **25.52** | **25.11** | **24.52** | **23.54** | **22.97** | **22.62** | **22.35** |
| RRDB [134] | 28.79 | 23.66 | 23.72 | 23.68 | 23.29 | 22.75 | 22.32 | 22.08 | 21.83 | 21.40 |
| RRDB+SFM | **29.10** | **23.81** | **23.99** | **23.79** | **23.41** | **22.90** | **22.53** | **22.37** | **21.98** | **21.56** |
| ESRGAN [134] | 25.43 | 21.22 | 22.49 | 22.03 | 21.87 | 21.63 | 21.21 | 20.99 | 20.05 | 19.42 |
| ESRGAN+SFM | **25.50** | **21.37** | **22.78** | **22.26** | **22.08** | **21.80** | **21.33** | **21.10** | **20.13** | **19.77** |

Table 2.1 – Single-image SR, with x4 upscaling factor, PSNR ($dB$) results on the DIV2K validation set. RCAN, RRDB and ESRGAN are trained using bicubic degradation, and IKC using Gaussian kernels ($\sigma \in [2.0, 4.0]$). Kernels seen in training are shaded gray. The training setups of the networks are presented in Sec. 2.5.1, and identical ones are used with SFM. We note that SFM improves the results of the various methods, even the IKC method that explicitly models kernels during its training improves by up to $0.27dB$ with SFM on unseen kernels.

When training for additive white Gaussian noise (AWGN) removal, we apply SFM on the clean image before the synthetic noise is added. When the training images are real and the noise cannot be separated from the signal, we apply SFM on the noisy image. Hence, we ensure that networks trained with SFM do not utilize any additional training data relative to the baselines. In all denoising experiments, and for all of the compared methods, we use *the same* SFM settings. We apply SFM on 50% of training images, and use the *targeted mode* of our SFM (ablation studies including other rates are in our *Supplementary Material*). We use a central band $r_C = 0.85\, r_M$ and $\sigma_\delta = 0.15\, r_M$. As presented in Section 2.4.1, this means that the highest frequency bands are masked with high likelihood, and lower frequencies are exponentially less likely to be masked the smaller they are. We add SFM to the training of the original methods, with no other modification.

## 2.5 Experiments

### 2.5.1 SR: Bicubic and Gaussian Degradations

**Methods**. We evaluate our proposed SFM method on state-of-the-art SR networks that can be divided into three categories. In the first category, we evaluate RCAN [159] and RRDB [134]; they are networks that target pixel-wise distortion for a single degradation kernel. RCAN employs a residual-in-residual structure and channel attention for efficient non-blind SR learning. RRDB [134] employs a residual-in-residual dense block as its basic architecture unit. The second category covers perception-optimized methods for a single degradation kernel and includes ESRGAN [134]. It is a version of the RRDB network that uses a GAN for better SR perceptual quality and obtains the state-of-the-art results in this category. The last category includes algorithms for blind SR. We experiment on IKC [54], which incorporates into the training of the SR network a blur-kernel estimation and modeling to explicitly address blind SR.

**Setup**. We train all the models by using the DIV2K [1] dataset. It is a high-quality dataset commonly used

| | Dataset and upscaling factor | | | | |
|---|---|---|---|---|---|
| | RealSR [19] | | | SR-RAW [157] | |
| Method | x2 | x3 | x4 | x4 | x8 |
| RCAN‡ [159] | 33.24 | 30.24 | 28.65 | 26.29 | 24.18 |
| RCAN 50% SFM | **33.32** | **30.29** | **28.75** | **26.42** | **24.50** |
| KMSR [164] | 32.98 | 30.05 | 28.27 | 25.91 | 24.00 |
| KMSR 50% SFM | **33.21** | **30.11** | **28.50** | **26.19** | **24.31** |
| IKC [54] | 33.07 | 30.03 | 28.29 | 25.87 | 24.19 |
| IKC 50% SFM | **33.12** | **30.25** | **28.42** | **25.93** | **24.25** |

Table 2.2 – PSNR ($dB$) results of blind image super-resolution on two real SR datasets, for the different available upscaling factors. ‡RCAN is trained on the paired dataset collected from the same sensor as the testing dataset.

for single-image SR evaluation. RCAN, RRDB, and ESRGAN are trained with the bicubic degradation; and IKC with Gaussian kernels ($\sigma \in [0.2, 4.0]$ [54]). For all models, 16 LR patches of size $48 \times 48$ are extracted per training batch. All models are trained using the Adam optimizer [70] for 50 epochs. The initial learning rate is set to $10^{-4}$ and decreases by half every 10 epochs. Data augmentation is performed on the training images that are randomly rotated by $90°$, $180°$, $270°$, and flipped horizontally.

**Results**. To generate test LR images, we apply bicubic and Gaussian blur kernels on the DIV2K [1] validation set. We also evaluate all methods trained with 50% SFM, following Section 2.4.2. Table 2.1 shows the PSNR results on x4 upscaling SR, with different blur kernels. Results show that the proposed SFM consistently improves the performance of the various SR networks on the different degradation kernels, even up to $0.27dB$ on an unseen test kernel for the recent IKC [54] that explicitly models kernels during training. We improve by up to $0.56dB$ for the other methods. With SFM, RRDB achieves comparable or better results than RCAN, while RCAN has double the parameters of RRDB. Sample visual results are shown in Figure 2.5.

### 2.5.2 SR: Real-Image Degradations

**Methods**. We train and evaluate the same SR models as the networks we use in Section 2.5.1, except for ESRGAN and RRDB; because ESRGAN is a perceptual-quality-driven method and does not achieve high PSNR, and RCAN outperforms RRDB according to our experiments in 2.5.1. We also evaluate on KMSR [164] for the real SR experiments. KMSR collects real blur kernels from real LR images to improve the generalization of the SR network on unseen kernels.

**Setup**. We train and evaluate the SR networks on two digital zoom datasets: the SR-RAW dataset [157] and the RealSR dataset [19]. The training setup of the SR networks is the same as in Section 2.5.1. Note that we follow the same training procedures for each method as in the original papers. IKC is trained with Gaussian kernels ($\sigma \in [0.2, 4.0]$) and KMSR with the blur kernels estimated from LR images in the dataset. RCAN is trained *on the degradation of the test* data; a starting advantage over other methods.

**Results**. We evalute the SR methods on the corresponding datasets and present the results in Table 2.2. Each method is also trained with 50% SFM, following Section 2.4.2. SFM consistently improves all methods

| | Test noise level (standard deviation of the stationary AWGN) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| DnCNN-B [150] | 33.33 | 29.71 | 27.66 | 26.13 | 24.88 | 23.69 | 22.06 | 19.86 | 17.88 | 16.35 |
| DnCNN-B + SFM | **33.35** | **29.78** | **27.73** | **26.27** | **25.09** | **24.02** | **22.80** | **21.24** | **19.46** | **17.87** |
| Noise2Noise [80] | **32.67** | 28.84 | 26.61 | 25.02 | 23.76 | 22.69 | 21.74 | 20.88 | 20.11 | 19.41 |
| Noise2Noise + SFM | 32.55 | **28.94** | **26.84** | **25.31** | **24.11** | **23.05** | **22.14** | **21.32** | **20.61** | **19.95** |
| Blind‡ N3Net [103] | **33.53** | **30.01** | **27.84** | 26.30 | 25.04 | 23.93 | 22.87 | 21.84 | 20.87 | 19.98 |
| N3Net + SFM | 33.41 | 29.86 | **27.84** | **26.38** | **25.19** | **24.15** | **23.20** | **22.32** | **21.51** | **20.78** |
| Blind‡ MemNet [125] | **33.51** | 29.75 | 27.61 | 26.06 | 24.87 | 23.83 | 22.67 | 21.00 | 18.92 | 17.16 |
| MemNet + SFM | 33.36 | **29.80** | **27.76** | **26.31** | **25.14** | **24.09** | **23.09** | **22.00** | **20.77** | **19.46** |
| RIDNet [6] | **33.65** | **29.87** | 27.65 | 26.04 | 24.79 | 23.65 | 22.25 | 20.05 | 18.15 | 17.09 |
| RIDNet + SFM | 33.43 | 29.81 | **27.76** | **26.30** | **25.12** | **24.08** | **23.11** | **22.08** | **20.74** | **19.17** |

Table 2.3 – PSNR ($dB$) results on BSD68 for different methods and noise levels. SFM improves the various methods, and the improvement increases with higher noise levels, supporting our hypothesis. We clamp test images to [0,255] as in camera pipelines. Denoisers are trained with levels up to 55 (shaded in gray), thus half the test range is not seen in training. ‡Re-trained under blind settings.

on all upscaling factors, pushing the state-of-the-art results by up to $0.23dB$ on both of these challenging real-image SR datasets.

### 2.5.3 Denoising: AWGN

**Methods**. We evaluate different state-of-the-art AWGN denoisers. DnCNN-B [150] learns the noise residual rather than the final denoised image. Noise2Noise (N2N) [80] learns only from noisy image pairs, with no ground-truth data. N3Net [103] relies on learning nearest neighbors similarity, to make use of different similar patches in an image for denoising. MemNet [125] follows residual learning with memory transition blocks. Lastly, RIDNet [6] also does residual learning, but uses feature attention blocks.

**Setup**. We train all methods on the 400 Berkeley images [91], typically used to benchmark denoisers [24, 112, 150]. All methods use the Adam optimizer with an initial learning rate of $10^{-3}$, except for RIDNet that uses half that rate. We train for 50 epochs and synthesize noise instances per training batch. For blind denoising training, we follow the settings initially set in [150]: noise is sampled from a Gaussian distribution with standard deviation chosen at random in $[0, 55]$. This splits the range of test noise levels into levels seen or not seen during training, which provides further insights on generalization. We also note that we use a U-Net [107] for the architecture of N2N as in the original work. For N2N, we apply SFM on top of the added noise, to preserve the particularity that N2N can be trained without ground-truth data.

**Results**. We evaluate all methods on the BSD68 [108] test set. Each method is also trained with 50% SFM as explained in Section 2.4.2 and the results are in Table 2.3. SFM improves the performance of a variety of different state-of-the-art denoising methods on high noise levels (seen during training, such as 40 and 50, or not even seen), and the results support our hypothesis presented in Section 2.3.2 that *the higher the noise level* the more similar denoising is to SR and the more applicable SFM is. Indeed, the higher the noise level the larger the improvement of SFM, and this trend is true across all methods. Figure 2.6 presents sample

| | # raw images for averaging | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Mixed test set [158] | | | | | Two-photon test set [158] | | | | |
| Method | 16 | 8 | 4 | 2 | 1 | 16 | 8 | 4 | 2 | 1 |
| PURE-LET [86] | 39.59 | 37.25 | 35.29 | 33.49 | 31.95 | 37.06 | 34.66 | 33.50 | 32.61 | 31.89 |
| VST+KSVD [2] | 40.36 | 37.79 | 35.84 | 33.69 | 32.02 | 38.01 | 35.31 | 34.02 | 32.95 | 31.91 |
| VST+WNNM [56] | 40.45 | 37.95 | 36.04 | 34.04 | 32.52 | 38.03 | 35.41 | 34.19 | 33.24 | 32.35 |
| VST+BM3D [29] | 40.61 | 38.01 | 36.05 | 34.09 | 32.71 | 38.24 | 35.49 | 34.25 | 33.33 | 32.48 |
| VST+EPLL [165] | 40.83 | 38.12 | 36.08 | 34.07 | 32.61 | 38.55 | 35.66 | 34.35 | 33.37 | 32.45 |
| N2S [9] | 36.67 | 35.47 | 34.66 | 33.15 | 31.87 | 34.88 | 33.48 | 32.66 | 31.81 | 30.51 |
| N2S 50% SFM | 36.60 | 35.62 | 34.59 | 33.44 | 32.40 | 34.39 | 33.14 | 32.48 | 31.84 | 30.92 |
| N2N [80] | 41.45 | 39.43 | 37.59 | 36.40 | 35.40 | 38.37 | 35.82 | 34.56 | 33.58 | 32.70 |
| N2N 50% SFM | **41.48** | **39.46** | **37.78** | **36.43** | **35.50** | **38.78** | **36.10** | **34.85** | **33.90** | **33.05** |

Table 2.4 – PSNR ($dB$) results on microscopy images with Poisson-Gaussian noise. We train under blind settings and apply SFM on noisy input images to preserve the fact that N2S and N2N can be trained without clean images.

results.

(a) Noisy (b) DnCNN (c) N2N (d) N3Net (e) MemNet (f) RIDNet

(g) GT (h) +SFM (i) +SFM (j) +SFM (k) +SFM (l) +SFM

Figure 2.6 – Denoising ($\sigma = 50$) results with different methods (top row), and with our SFM added (bottom row), for the last image (#67) of the BSD68 benchmark.

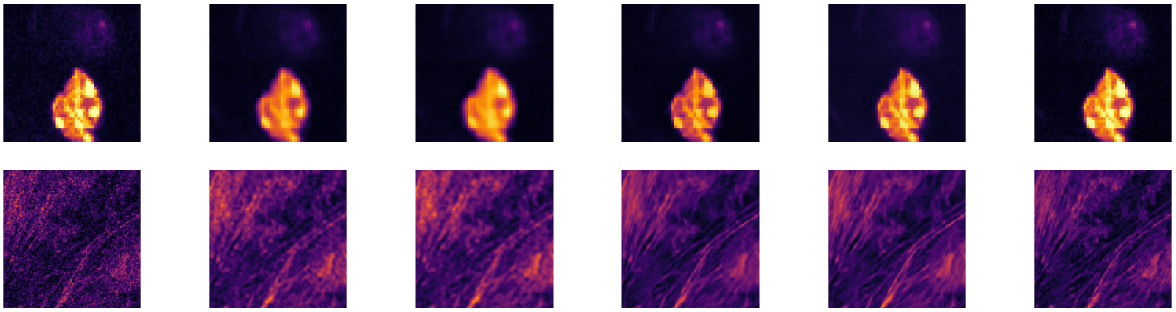
### 2.5.4 Denoising: Real Poisson-Gaussian Images

**Methods**. In the absence of ground-truth datasets, classic methods are often a good choice for denoising. PURE-LET [86] is specifically aimed at Poisson-Gaussian denoising, and KSVD [2], WNNM [56], BM3D [29], and EPLL [165] are designed for Gaussian denoising. Recently, learning methods were presented, such as N2S [9] (and the similar, but less general, N2V [72]) that can learn from a dataset of only noisy images, and N2N [80] that can learn from a dataset of only noisy image pairs. We incorporate SFM into the learning-based methods.

**Setup**. We train the learning-based methods on the recent real fluorescence microscopy dataset [158]. The

noise follows a Poisson-Gaussian distribution, and the image registration is of high quality due to the stability of the microscopes, hence yielding reliable ground truth obtained by averaging 50 repeated captures. Noise parameters are estimated using the fitting approach in [50] for all classic denoisers. Additionally, the parameters are used for the variance-stabilization transform (VST) [89] for the Gaussian-oriented methods. In contrast, the learning methods can directly be applied under blind settings. We train N2S/N2N using a U-Net [107] architecture, for 100/400 epochs using the Adam optimizer with a starting learning rate of $10^{-5}$/$10^{-4}$ [158].

**Results**. We evaluate on the mixed and two-photon microscopy test sets [158]. We also train the learning methods with 50% SFM as explained in Section 2.4.2, and we present the results in Table 2.4. A larger number of averaged raw images is equivalent to a lower noise level. N2N with SFM achieves the state-of-the-art performance on both benchmarks and for all noise levels, with an improvement of up to $0.42dB$. We also note that the improvements of SFM are larger on the more challenging two-photon test set where the noise levels are higher on average. SFM does not consistently improve N2S, however, this is expected. In fact, unlike other methods, N2S trains to predict a subset of an image given a surrounding subset. It applies spatial masking where the mask is made up of random pixels and interferes with the frequency components. For these reasons, N2S is not very compatible with SFM that, nonetheless, improves results on the largest noise levels in both test sets.



(a) Noisy (b) N2S (c) + SFM (d) N2N (e) + SFM (f) GT

Figure 2.7 – Cropped sample results for denoising image (a) from the real fluorescence microscopy denoising dataset. The top row averages 16 raw images (MICE scan) to obtain (a), and the bottom row directly denoises from 1 image only (BPAE scan). The 'ground-truth' image (f) is estimated by averaging 50 raw images [158].

## 2.6 Ablation Studies

### 2.6.1 Super-Resolution

#### Varying Masked Bands

This ablation study investigates the effect of the frequency-domain masking on SR when applied on different *targeted* narrow frequency bands rather than on the wide-band frequency masking carried out by the *central mode* of SFM. We test on two different frequency bands, namely Low-Frequency Masking (LFM), in which the target band is set to $r_C = 0.25\, r_M$, and High-Frequency Masking (HFM), in which the target band is

| | Test blur kernel ($g_\sigma$ is a Gaussian kernel, standard deviation $\sigma$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bicubic | $g_{1.7}$ | $g_{2.3}$ | $g_{2.9}$ | $g_{3.5}$ | $g_{4.1}$ | $g_{4.7}$ | $g_{5.3}$ | $g_{5.9}$ | $g_{6.5}$ |
| RCAN [159] | 29.18 | 23.80 | 24.08 | 23.76 | 23.35 | 22.98 | 22.38 | 22.16 | 21.86 | 21.72 |
| RCAN 50% LFM | 29.23 | 23.91 | 24.50 | 24.12 | 23.70 | **23.29** | 22.53 | **22.25** | **21.91** | 21.77 |
| RCAN 50% SFM | **29.32** | **24.21** | **24.64** | **24.19** | **23.72** | 23.27 | **22.54** | 22.23 | **21.91** | **21.79** |
| RCAN 50% HFM | 29.20 | 23.78 | 24.07 | 23.79 | 23.39 | 22.98 | 22.40 | 22.17 | 21.88 | 21.75 |

Table 2.5 – PSNR (dB) results of blind image SR on the DIV2K validation set for RCAN, trained with different frequency-domain maskings, on different degradation kernels. Kernels seen in the training are shaded in gray. The proposed SFM outperforms not only the baseline but also the low-frequency masking (LFM) and high-frequency masking (HFM) on almost all the degradation kernels.

set to $r_C = 0.75\, r_M$. We control the average width of the band by setting $\sigma_\delta = 0.15\, r_M$, as in the *targeted mode* used for denoising.

We train RCAN [159] without any masking, with 50% SFM, with 50% LFM, and with 50% HFM on x4 SR on the DIV2K dataset by using the same experimental settings as earlier. Table 2.5 shows the PSNR results. We see that with SFM (and LFM), RCAN gains improvements on all the test degradation kernels. This further supports the premise that frequency-domain masking improves the learning of SR networks. SFM outperforms LFM and HFM on most of the degradation kernels, which shows the effectiveness of the proposed *central mode* frequency masking of SFM.

**Varying SFM Rates**

| | Test blur kernel ($g_\sigma$ is a Gaussian kernel, standard deviation $\sigma$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bicubic | $g_{1.7}$ | $g_{2.3}$ | $g_{2.9}$ | $g_{3.5}$ | $g_{4.1}$ | $g_{4.7}$ | $g_{5.3}$ | $g_{5.9}$ | $g_{6.5}$ |
| RCAN [159] | 29.18 | 23.80 | 24.08 | 23.76 | 23.35 | 22.98 | 22.38 | 22.16 | 21.86 | 21.72 |
| RCAN 25% SFM | **29.35** | 24.18 | 24.59 | **24.21** | 23.67 | 23.25 | 22.48 | **22.31** | 21.90 | 21.78 |
| RCAN 50% SFM | 29.32 | **24.21** | **24.64** | 24.19 | **23.72** | **23.27** | **22.54** | 22.23 | 21.91 | 21.79 |
| RCAN 75% SFM | 29.21 | 24.02 | 24.32 | 24.04 | 23.62 | 23.17 | 22.46 | 22.24 | **21.95** | 21.82 |
| RCAN 100% SFM | 29.28 | 23.78 | 24.11 | 23.69 | 23.44 | 23.05 | 22.41 | 22.25 | 21.93 | **21.85** |

Table 2.6 – PSNR (dB) results of blind image SR on the DIV2K validation set for RCAN on different degradation kernels. We present an ablation study over different rates of SFM. Note that 100% SFM means we mask every training input using SFM. Kernels seen in the training are shaded in gray. The results show that with any rates between 25% and 75%, SFM improves the performance of the SR network on all the degradation kernels.

This ablation study investigates the effect of varying the rate of SFM that is applied during training. We train RCAN [134] on x4 SR with a varying percentage of patches being masked with SFM. The results are shown in Table 2.6. With 25% and 50% SFM, we achieve the best performance on most of the degradation kernels. The network trained with rates [25, 50, 75]% SFM outperforms the baseline method under all test degradation kernels, and even the one with 100% masking outperforms the baseline on all test kernels except for two. This shows that our proposed SFM is effective with different rates and is not very sensitive to the

chosen percentage of masked training patches.

### 2.6.2 Denoising

#### Low-Frequency Masking

In this ablation study, we investigate the effect of the frequency-domain masking when it is applied on low-frequency components rather than the high-frequency masking carried out by SFM. We use the same name as in the SR experiments and call this masking LFM, for low-frrequency masking, although the target frequency is smaller, as described next. The same denoiser training pipeline is used, with the same approach for applying the masking, except that the central band is set to $r_C = 0.15 * r_M$ rather than $r_C = 0.85 * r_M$ as in our SFM.

| | Test noise level (standard deviation of the stationary AWGN) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| DnCNN-B [150] | 33.33 | 29.71 | 27.66 | 26.13 | 24.88 | 23.69 | 22.06 | 19.86 | 17.88 | 16.35 |
| DnCNN-B 50% LFM | 33.01 | 29.36 | 27.31 | 25.87 | 24.71 | 23.65 | 22.25 | 20.39 | 18.61 | 17.08 |
| DnCNN-B 50% SFM | **33.35** | **29.78** | **27.73** | **26.27** | **25.09** | **24.02** | **22.80** | **21.24** | **19.46** | **17.87** |
| Noise2Noise [80] | **32.67** | 28.84 | 26.61 | 25.02 | 23.76 | 22.69 | 21.74 | 20.88 | 20.11 | 19.41 |
| N2N 50% LFM | 27.32 | 26.15 | 25.16 | 24.21 | 23.34 | 22.57 | 21.83 | 21.10 | 20.40 | 19.73 |
| N2N 50% SFM | 32.55 | **28.94** | **26.84** | **25.31** | **24.11** | **23.05** | **22.14** | **21.32** | **20.61** | **19.95** |
| Blind* N3Net [103] | **33.53** | **30.01** | **27.84** | 26.30 | 25.04 | 23.93 | 22.87 | 21.84 | 20.87 | 19.98 |
| N3Net 50% LFM | 29.24 | 27.62 | 26.42 | 25.44 | 24.56 | 23.72 | 22.90 | 22.10 | 21.35 | 20.65 |
| N3Net 50% SFM | 33.41 | 29.86 | **27.84** | **26.38** | **25.19** | **24.15** | **23.20** | **22.32** | **21.51** | **20.78** |
| Blind* MemNet [125] | **33.51** | 29.75 | 27.61 | 26.06 | 24.87 | 23.83 | 22.67 | 21.00 | 18.92 | 17.16 |
| MemNet 50% LFM | 32.90 | 29.27 | 27.21 | 25.76 | 24.61 | 23.55 | 22.40 | 21.01 | 19.64 | 18.45 |
| MemNet 50% SFM | 33.36 | **29.80** | **27.76** | **26.31** | **25.14** | **24.09** | **23.09** | **22.00** | **20.77** | **19.46** |
| RIDNet [6] | **33.65** | **29.87** | 27.65 | 26.04 | 24.79 | 23.65 | 22.25 | 20.05 | 18.15 | 17.09 |
| RIDNet 50% LFM | 31.48 | 28.06 | 25.99 | 24.55 | 23.45 | 22.52 | 21.70 | 20.96 | 20.27 | **19.65** |
| RIDNet 50% SFM | 33.43 | 29.81 | **27.76** | **26.30** | **25.12** | **24.08** | **23.11** | **22.08** | **20.74** | 19.17 |

Table 2.7 – PSNR ($dB$) results of blind AWGN image denoising on the standard BSD68 test set for different methods and noise levels. SFM improves the various methods, and the improvement increases with increasing noise levels, validating our hypothesis. We clamp noisy test images to [0,255] as in camera pipelines, to follow practical settings. LFM stands for low-frequency masking, which is similar to applying SFM but on low-frequency components rather than high-frequency ones, i.e. opposite to our proposed SFM approach. *We re-train under blind noise settings. The gray background indicates noise levels seen during training.

In Table 2.7, we present the results on additive white Gaussian noise removal, without any masking, with 50% SFM, and with 50% LFM. The results are given for the various denoising methods. The masking of low-frequency components is always worse than the high-frequency masking of SFM (the only exception is the RIDNet at noise level 100). LFM is almost always worse than the baseline, with some exceptions when the noise level is significantly high and even relatively low-frequency components are actually overtaken by the additive noise. The results show the importance of not simply masking any frequency components but specifically high-frequency ones.

**Varying SFM Rates**

In this ablation study, we investigate the effect of varying the rate of SFM that is applied during training. In other words, we vary the percentage of total training patches that are masked by using SFM and we analyze the resulting performances. We conduct this ablation study on the state-of-the-art method, namely N2N [80], on the real fluorescence microscopy image benchmark test sets [158]. All training settings follow exactly the description, with the only variable being the percentage of SFM-masked training patches. We repeat the training with 0, 10, 25, 50, 75, 90 and 100% of images masked by using SFM.

| | # raw images for averaging | | | | |
|---|---|---|---|---|---|
| Method | **16** | **8** | **4** | **2** | **1** |
| | Mixed test set [158] | | | | |
| N2N [80] | 41.45 | 39.43 | 37.59 | 36.40 | 35.40 |
| N2N 10% SFM | 41.35 | 39.35 | 37.73 | 36.32 | 35.45 |
| N2N 25% SFM | **41.50** | 39.45 | **37.79** | 36.41 | **35.52** |
| N2N 50% SFM | 41.48 | **39.46** | 37.78 | 36.43 | 35.50 |
| N2N 75% SFM | 41.40 | 39.44 | 37.75 | 36.46 | 35.50 |
| N2N 90% SFM | 41.38 | **39.46** | 37.76 | **36.47** | 35.50 |
| N2N 100% SFM | 41.25 | 39.40 | 37.70 | 36.43 | 35.45 |
| | Two-photon test set [158] | | | | |
| N2N [80] | 38.37 | 35.82 | 34.56 | 33.58 | 32.70 |
| N2N 10% SFM | 38.68 | 35.98 | 34.79 | 33.90 | 33.03 |
| N2N 25% SFM | **38.81** | 36.06 | 34.84 | **33.95** | **33.10** |
| N2N 50% SFM | 38.78 | **36.10** | **34.85** | 33.90 | 33.05 |
| N2N 75% SFM | 38.71 | 36.02 | 34.76 | 33.81 | 33.01 |
| N2N 90% SFM | 38.69 | 36.07 | 34.80 | 33.87 | 33.05 |
| N2N 100% SFM | 38.39 | 35.78 | 34.52 | 33.61 | 32.84 |

Table 2.8 – PSNR ($dB$) denoising results on real fluorescence microscopy images with Poisson-Gaussian noise. We present an ablation study over different rates of SFM. Note that 100% SFM means we mask every training input image using SFM. We highlight with gray background the results that *do not* outperform the previous state of the art on both benchmark datasets. Results confirm that even with very small (10%), or with very extreme SFM rates (100%), using SFM improves the results on the high noise level scenarios where our theory is most applicable.

We present the results of the real-image denoising task on the mixed microscopy test set and the two-photon test set [158] in Table 2.8. The top results are distributed between 25 and 90% SFM, with the best ones being usually at 25 or 50% SFM rates. The gray background highlights the cases where N2N with a certain SFM rate does not improve the baseline. These few exceptions are either at extreme SFM rates (10 or 100%) or at very low noise levels (the lowest one, in fact). This further validates the theoretical proposition we make that SFM becomes more applicable when the noise level is higher. The results also show that they are not very sensitive with respect to the SFM rate, as all SFM models with rates in [25, 90]% outperform the previous state-of-the-art method on almost all noise levels in both test sets.

## 2.7 Extended Experimental Evaluation

### 2.7.1 Super-Resolution

#### Different Upscaling Factors

| | Test blur kernel ($g_\sigma$ is a Gaussian kernel, standard deviation $\sigma$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bicubic | $g_{1.7}$ | $g_{2.3}$ | $g_{2.9}$ | $g_{3.5}$ | $g_{4.1}$ | $g_{4.7}$ | $g_{5.3}$ | $g_{5.9}$ | $g_{6.5}$ |
| RCAN [159] | 32.07 | 27.01 | 25.92 | 24.97 | 24.27 | 23.73 | 23.00 | 22.72 | 22.36 | 22.24 |
| RCAN 50% SFM | **32.20** | **27.19** | **26.21** | **25.35** | **24.63** | **24.10** | **23.38** | **22.91** | **22.44** | **22.29** |
| RRDB [134] | 31.93 | 27.00 | 25.95 | 24.83 | 24.16 | 23.69 | 22.89 | 22.65 | 22.19 | 22.13 |
| RRDB 50% SFM | **31.99** | **27.08** | **26.14** | **25.21** | **24.49** | **24.02** | **23.25** | **22.88** | **22.29** | **22.18** |
| ESRGAN [134] | 30.87 | 26.72 | 24.07 | 22.53 | 22.74 | 22.26 | 21.52 | 21.29 | 20.89 | 19.99 |
| ESRGAN 50% SFM | **30.90** | **26.81** | **24.25** | **22.66** | **22.94** | **22.49** | **21.78** | **21.40** | **21.95** | **20.03** |
| IKC [54] | **31.68** | **28.65** | 27.43 | 26.33 | 25.78 | 25.29 | 24.44 | 24.20 | 23.89 | 23.61 |
| IKC 50% SFM | 31.60 | 28.64 | **27.51** | **26.46** | **25.99** | **25.42** | **24.67** | **24.51** | **24.08** | **23.79** |

Table 2.9 – Image SR PSNR ($dB$) results, with **x2 upscaling** factor, on the DIV2K validation set. Kernels seen in the trainng are shaded in gray. SFM improves the results of the various methods on different test blur kernels.

We present the PSNR results of RCAN [159], RRDB [134], ESRGAN [134] and IKC [54]; all were trained without and with SFM. The results with x2 and x8 upscaling factors are given, respectively, in Tables 2.9 and 2.10. We evaluate all the different degradation kernels analyzed earlier. We note that SFM improves the results of the various methods, on both SR upscaling factors, and on practically all degradation kernels, except the smallest Gaussian standard deviation ones for only IKC [54] that explicitly models and estimates all the *test* blur kernels *during training*.

#### DCT Evaluation

In this section, we analyze the reconstruction performance of x4 SR networks, trained with and without SFM, in the DCT frequency domain. We present the results of RCAN [159], ESRGAN [134] and IKC [54] in Figure 2.8, with one method per row. The first column shows the image PSNR improvement of models trained with SFM compared to the models trained without it, for different Gaussian blur kernels. The second and the third columns show the MSE (mean squared error) improvement on low-frequency and high-frequency components in the DCT domain. Low and high frequencies are split in the DCT domain by using an ideal frequency filtering with cutoff at $\pi/4$. We choose this separation cutoff rather than, for instance, $\pi/2$, because the SR network performs x4 upscaling. Hence, to avoid high-frequency aliasing, the anti-aliasing filter required before the downsampling must filter frequencies above $\pi/4$. Therefore, we adopt this definition for high- vs. low-frequency content.

The results show that SFM improves the reconstruction of SR networks for both low-frequency and high-frequency components (note that the scales are different because the typical image PSD is not uniform with respect to frequency, as discussed earlier). This improvement is consistent across the different test degradation

| | Test blur kernel ($g_\sigma$ is a Gaussian kernel, standard deviation $\sigma$) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | bicubic | $g_{1.7}$ | $g_{2.3}$ | $g_{2.9}$ | $g_{3.5}$ | $g_{4.1}$ | $g_{4.7}$ | $g_{5.3}$ | $g_{5.9}$ | $g_{6.5}$ |
| RCAN [159] | 22.67 | 21.49 | 21.61 | 21.80 | 21.86 | 21.86 | 21.51 | 21.51 | 21.45 | 21.20 |
| RCAN 50% SFM | **22.89** | **21.70** | **21.85** | **22.09** | **22.17** | **22.19** | **21.80** | **21.78** | **21.86** | **21.35** |
| RRDB [134] | 22.52 | 21.38 | 21.47 | 21.58 | 21.63 | 21.65 | 21.30 | 21.26 | 21.14 | 21.08 |
| RRDB 50% SFM | **22.59** | **21.45** | **21.56** | **21.80** | **21.82** | **21.87** | **21.53** | **21.48** | **21.37** | **21.18** |
| ESRGAN [134] | 21.64 | 18.94 | 19.16 | 19.35 | 19.63 | 19.72 | 19.12 | 19.08 | 19.01 | 18.97 |
| ESRGAN 50% SFM | **21.92** | **19.04** | **19.37** | **19.62** | **19.87** | **19.99** | **19.36** | **19.31** | **19.29** | **19.15** |
| IKC [54] | **22.33** | **22.64** | **22.87** | 22.93 | 23.02 | 22.87 | 22.65 | 22.61 | 22.58 | 22.33 |
| IKC 50% SFM | 22.28 | 22.58 | 22.84 | **22.97** | **23.09** | **23.01** | **22.78** | **22.73** | **22.69** | **22.50** |

Table 2.10 – Image SR PSNR ($dB$) results, with **x8 upscaling** factor, on the DIV2K validation set. Kernels seen in the training are shaded in gray. SFM improves the results of the various methods on different test blur kernels.

kernels and SR methods. The results also show that the improvement on reconstructing high-frequency components does not come at the expense of the low-frequency reconstruction.

#### Visual Results: Synthetic Kernels

We present more visual results of synthetic x4 SR. We show the results of RCAN [159], ESRGAN [134] and IKC [54]; all were trained with and without 50% SFM, with different degradation kernels, in Figure 2.9, 2.10, 2.11, and 2.12. For each of these methods, in the bottom row, we show the results of the same method trained with the same settings and starting from the same network initialization, but we use our proposed SFM with a 50% rate. With SFM, the SR networks are able to produce sharper results.

#### Visual Results: Real Datasets

We present more visual results from real SR datasets, in Fig 2.13, 2.14 and 2.15. We show the SR results of RCAN [159], KMSR [164] and IKC [54]. For each of these three methods, we also show the results of the same version trained with 50% SFM. We clearly see that SFM improves the visual quality of the SR networks' results.

### 2.7.2 Denoising

#### DCT Evaluation

So far, we have evaluated the performance of the trained denoisers, with and without SFM, by using the standard PSNR metric. In this section, we are interested in analyzing the reconstruction performance in the DCT frequency domain. Figure 2.16 shows results with different methods, one method per row. The first column shows the image PSNR improvement of methods trained with SFM relative to without it, for every noise level in the range 10 to 100, with steps of 10. The second and third columns also show the

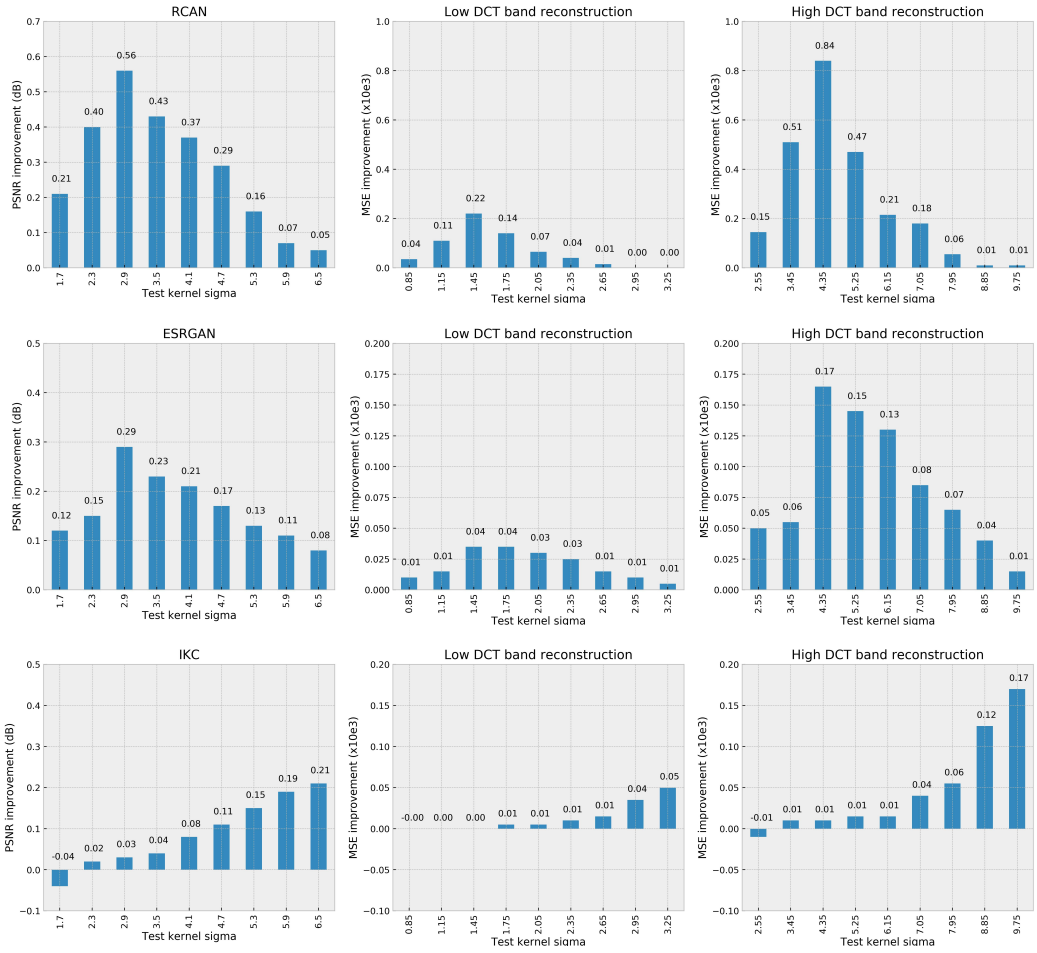
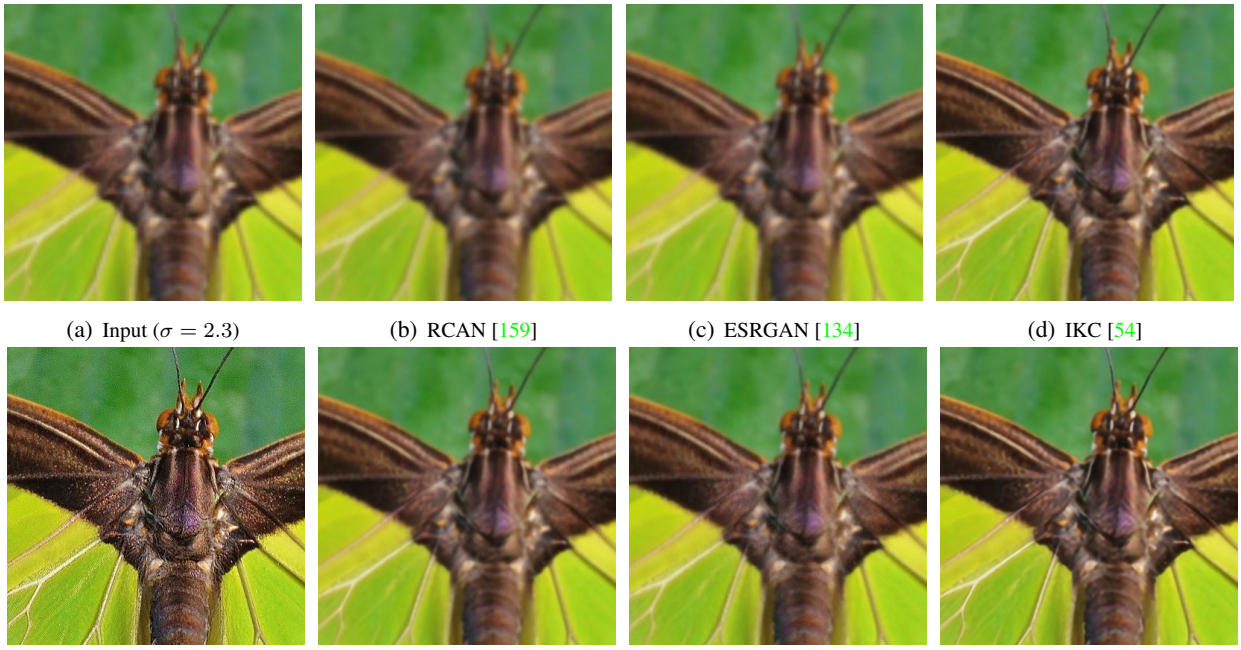



Figure 2.8 – The first column shows the PSNR improvements due to SFM for blind SR on the DIV2K for varying degradations. The second and third columns show the improvements in MSE computed, respectively, on low and high frequencies. The results show that the improvements obtained on reconstructing the high-frequency content do not come at the cost of low-frequency content reconstruction; on the contrary, both are improved.

improvement per noise level but are evaluated in the DCT domain. In the second column, we show the improvement with SFM in terms of MSE computed on the low frequencies of the image. Similarly, in the third column, we show the MSE improvement of using SFM but on the high-frequency components. Low and high frequencies are split in the DCT domain into two equal ranges, thus simulating an ideal frequency filtering with cutoff at $\pi/2$.

The results illustrate the increase in improvement as the noise level increases, hence supporting our hypothesis. Furthermore, we see that the improvement in reconstruction is notable in both low and high frequencies, for the different methods. SFM does indeed improve the reconstruction of high frequencies by forcing the network during training to predict them from their low-frequency counterpart (corresponding to the bottom conditional distribution in Equation (2.8)). Also importantly, this procedure is not damaging the denoising of low frequencies (corresponding to the top conditional distribution in Equation (2.8)), as shown



(a) Input ($\sigma = 2.3$) (b) RCAN [159] (c) ESRGAN [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) ESRGAN + SFM (h) IKC + SFM

Figure 2.9 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image 0829 of the DIV2K benchmark.

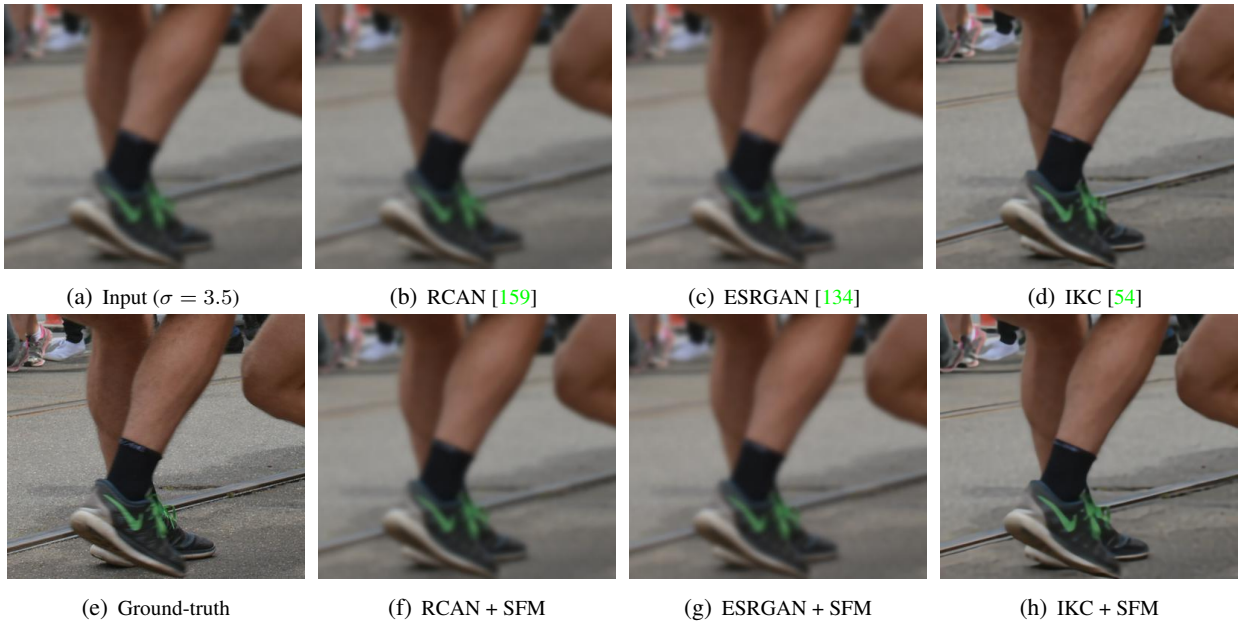
by the results in the third column. Our SFM even improves the reconstruction of those low frequencies, possibly because the network has a direct view of them during training when SFM masks the high-frequency counterpart in the input.

### Visual Results: AWGN

We present visual denoising results for additive white Gaussian noise removal. We show the results of DnCNN [150], N2N [80], N3Net [103], MemNet [125], and RIDNet [6]; all are on different images from the BSD68 benchmark, for different noise levels. For each of these methods, in the bottom row, we also show the results of the same method trained with the same settings and starting from the same network initialization, but we use our proposed SFM with a 50% rate. The results are shown in Figure 2.17, 2.18, 2.19, 2.20, 2.21, 2.22, 2.23, 2.24, and 2.25. The first column shows the noisy input image (and the corresponding standard deviation of the AWGN) in the top row, and the ground-truth image in the bottom row.

### Visual Results: Real Poisson-Gaussian Images

We present visual denoising results from the real-image fluorescence microscopy dataset in Figure 2.26, 2.27, and 2.28. The first column shows the noisy images obtained by averaging a different number of raw images to indirectly control the noise level; and the last column shows that the ground-truth images are estimated



(a) Input ($\sigma = 3.5$) (b) RCAN [159] (c) ESRGAN [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) ESRGAN + SFM (h) IKC + SFM

Figure 2.10 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image 0832 of the DIV2K benchmark.

by averaging 50 raw images for every scan. We present the results of the two methods that can be trained without ground-truth data for such real-image datasets, namely, N2S [9] and N2N [80]. For each of these two methods, we also show the results of the same version trained with 50% SFM.

## 2.8 Conclusion

In this chapter, we have analyzed the degradation-kernel overfitting of SR networks in the frequency domain. Our frequency-domain analysis reveals an implicit conditional learning that also extends to denoising, especially on high noise levels. This highlighted form of a learned prior not only provides extended understanding of the inner workings of the networks but also enables us to improve their learning. Building on our analysis, we present SFM, a technique for improving SR and denoising networks, without increasing the size of the training set and without incurring any cost at test time. We have conducted extensive experiments on state-of-the-art networks for both restoration tasks. We have evaluated SR with synthetic degradations, real-image SR, Gaussian denoising and with real-image Poisson-Gaussian denoising. We have shown improved performance, notably on generalization, when SFM is used.

One of the drawbacks of our SFM is that an additional processing of some of the training images is needed. To apply the SFM masking, the image is transformed to a frequency domain; in our case through the DCT, where the mask is applied, and is then transformed back. Although this has no effect at test time, it does increase the needed computations during the training phase. Another limitation is with the masks themselves. In SFM, our masks are limited to binary masks; this forms a spanning set of kernels. Although these masks need not be binary, a more important point is that our spanning set theoretically covers non-realistic kernels

(a) Input ($\sigma = 4.1$) (b) RCAN [159] (c) ESRGAN [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) ESRGAN + SFM (h) IKC + SFM

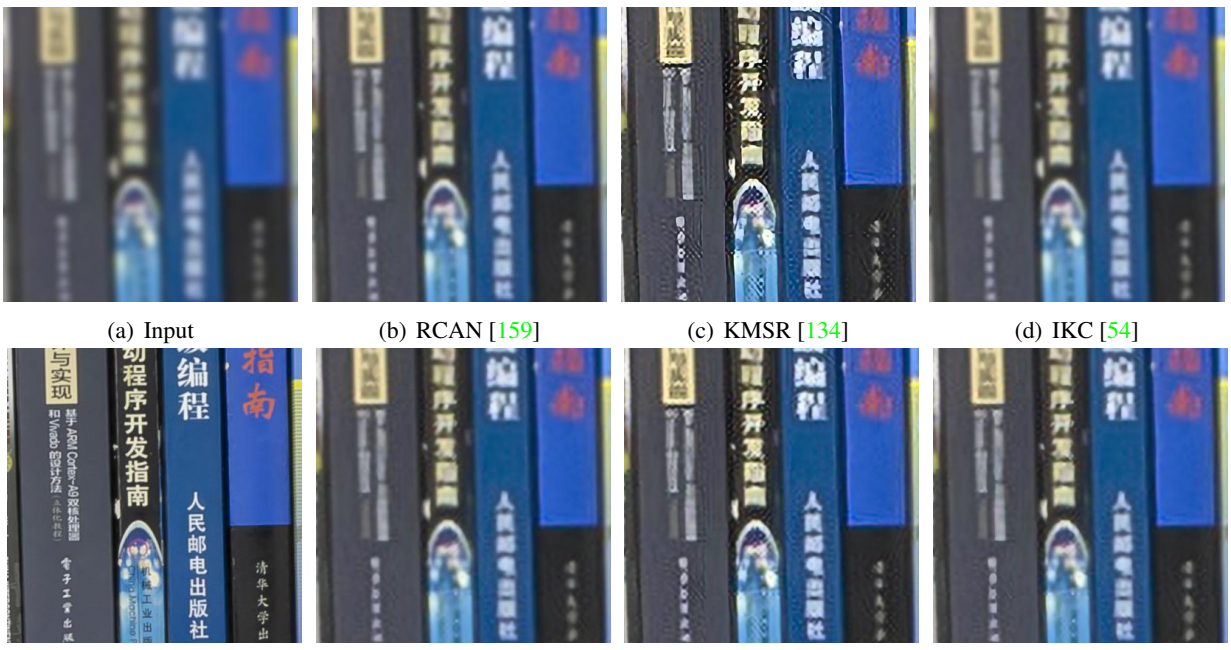
Figure 2.11 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image 0872 of the DIV2K benchmark.

or those that are not physically realizable. Limiting the simulated masks to span only a realistic set of kernels could potentially ease the learning and improve the performance of the networks. Furthermore, such masks could be applied in the spatial domain with convolutions, hence removing the need for the frequency-domain transformations.

(a) Input ($\sigma = 1.7$) (b) RCAN [159] (c) ESRGAN [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) ESRGAN + SFM (h) IKC + SFM

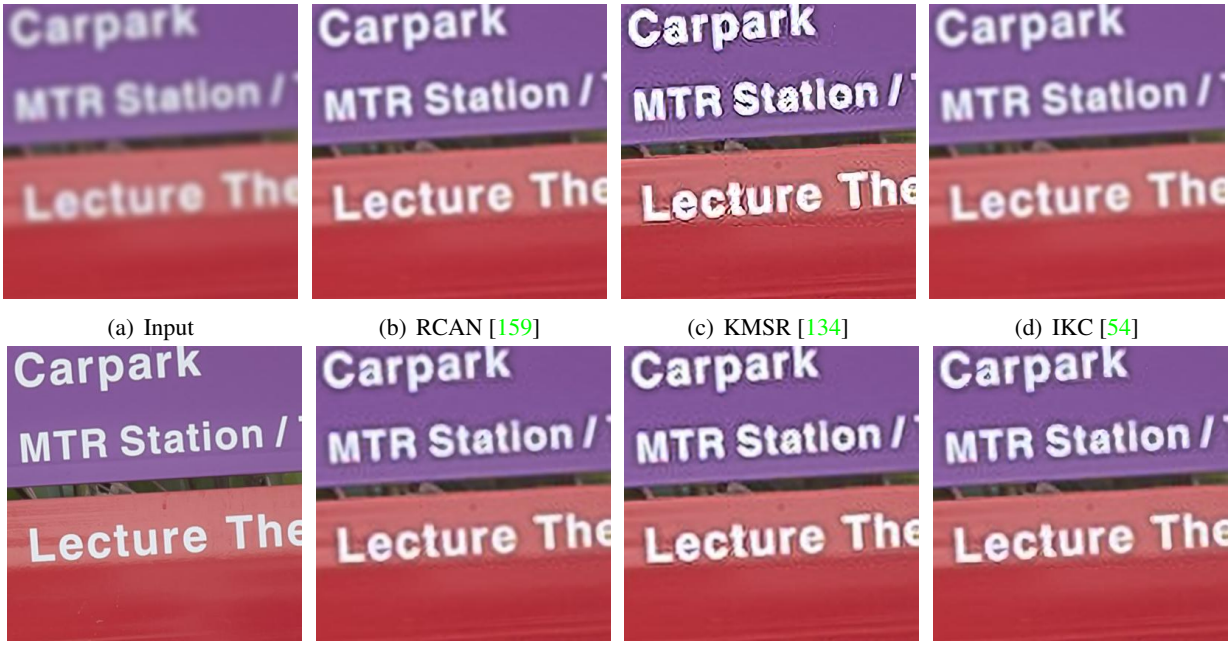
Figure 2.12 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image 0825 of the DIV2K benchmark.



(a) Input (b) RCAN [159] (c) KMSR [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) KMSR + SFM (h) IKC + SFM

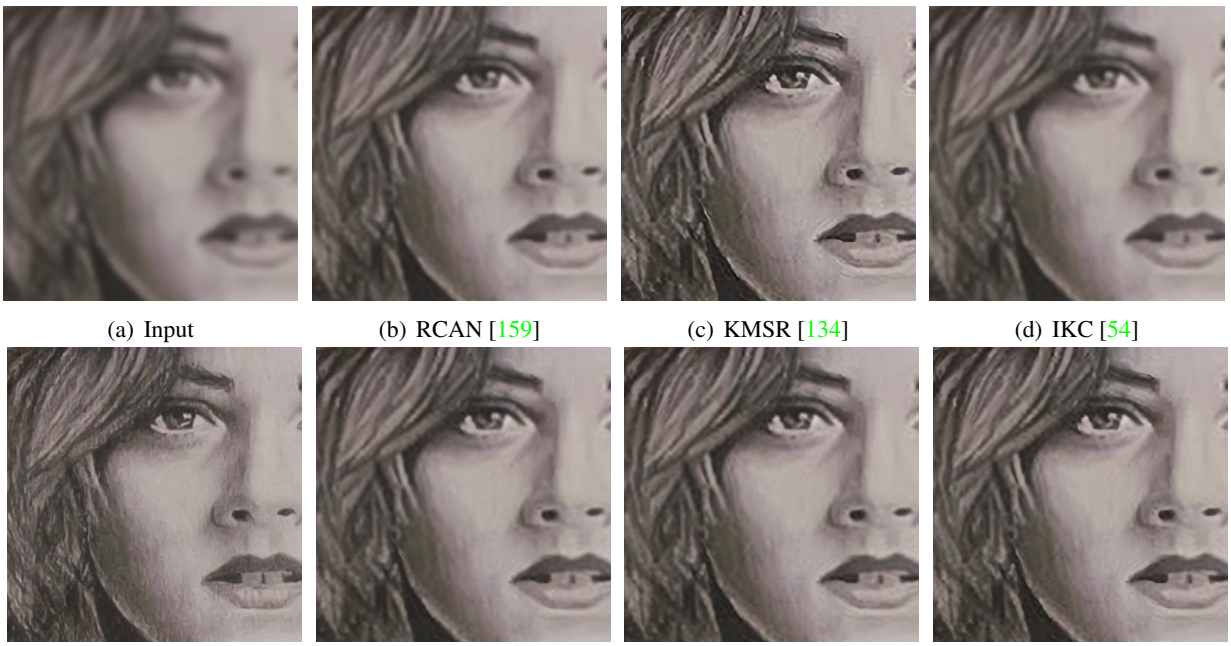
Figure 2.13 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image Canon_013 of the Real SR benchmark.




(a) Input (b) RCAN [159] (c) KMSR [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) KMSR + SFM (h) IKC + SFM

Figure 2.14 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image Nikon_004 of the Real SR benchmark.



(a) Input (b) RCAN [159] (c) KMSR [134] (d) IKC [54]

(e) Ground-truth (f) RCAN + SFM (g) KMSR + SFM (h) IKC + SFM

Figure 2.15 – Cropped SR results (x4 upscaling) with different methods (top row), and with the same methods trained with our SFM (bottom row), for image Nikon_011 of the Real SR benchmark.

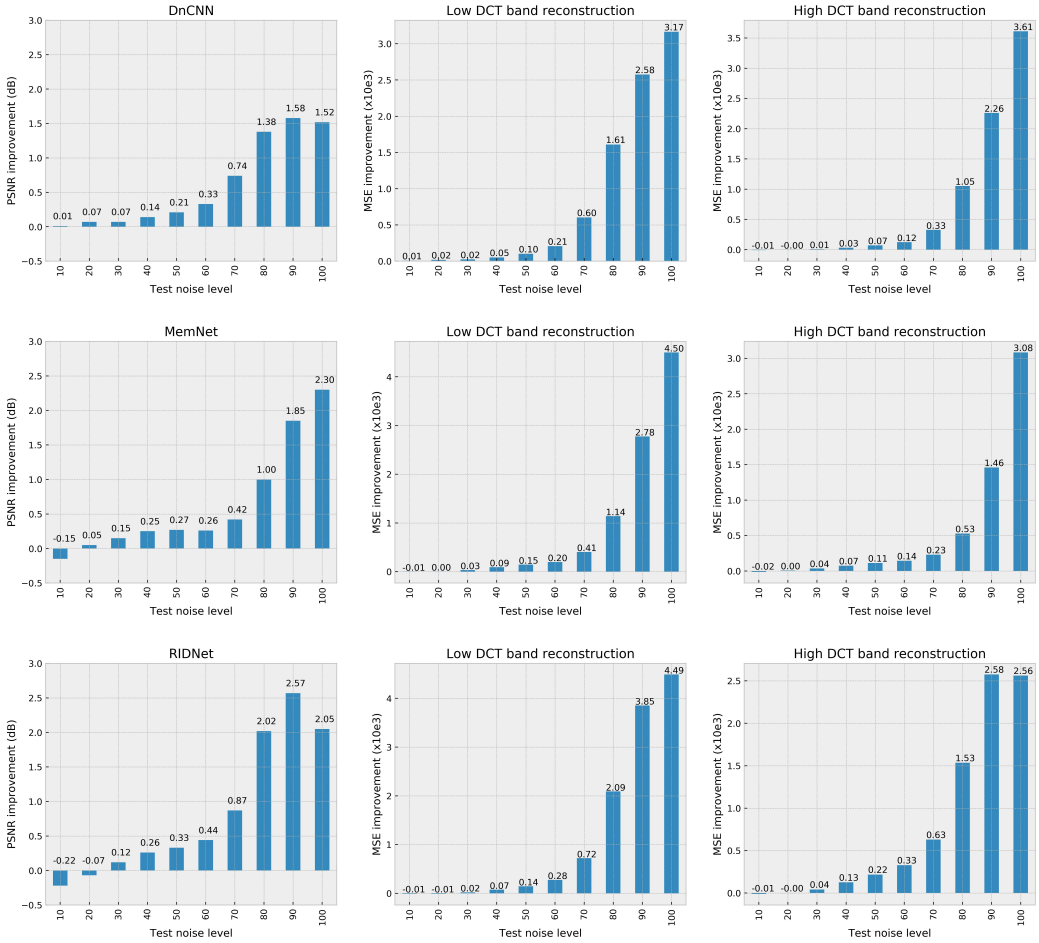
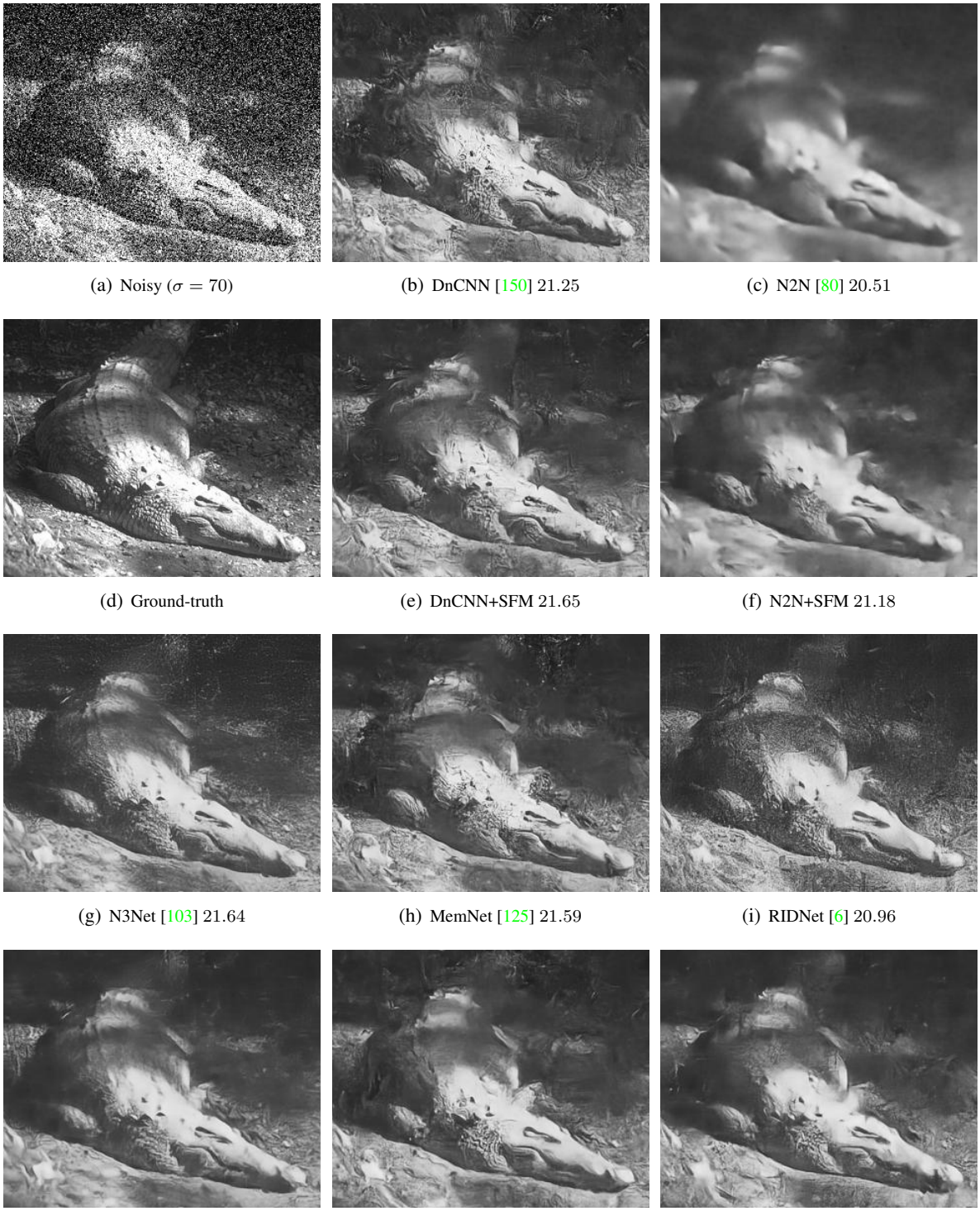



Figure 2.16 – The first column shows the PSNR improvements due to SFM for blind AWGN denoising on the BSD68 benchmark for varying noise levels from 10 to 100, with steps of 10. The second and third columns show the improvements in MSE computed, respectively, on low and high frequencies. The results show that the improvements obtained on reconstructing the high-frequency content does not come at the cost of low-frequency content reconstruction, on the contrary, both are improved. We also note that the improvement increases with increasing noise levels, supporting our original hypothesis.



(a) Noisy ($\sigma = 70$) (b) DnCNN [150] 21.25 (c) N2N [80] 20.51

(d) Ground-truth (e) DnCNN+SFM 21.65 (f) N2N+SFM 21.18

(g) N3Net [103] 21.64 (h) MemNet [125] 21.59 (i) RIDNet [6] 20.96

(j) N3Net+SFM 21.88 (k) MemNet+SFM 21.80 (l) RIDNet+SFM 21.88

Figure 2.17 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 14 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).

(a) Noisy ($\sigma = 80$) (b) DnCNN [150] 18.27 (c) N2N [80] 17.78 (d) N3Net [103] 18.93 (e) MemNet [125] 18.35 (f) RIDNet [6] 18.41

(g) Ground-truth (h) DnCNN+SFM 18.81 (i) N2N+SFM 18.48 (j) N3Net+SFM 19.22 (k) MemNet+SFM 19.06 (l) RIDNet+SFM 19.01

Figure 2.18 – Denoising results with different methods (top row), and with the same method trained with our SFM (bottom row), for image 20 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).

(a) Noisy ($\sigma = 60$) (b) DnCNN [150] 21.92 (c) N2N [80] 20.98

(d) Ground-truth (e) DnCNN+SFM 22.12 (f) N2N+SFM 21.42

(g) N3Net [103] 21.95 (h) MemNet [125] 21.74 (i) RIDNet [6] 21.98

(j) N3Net+SFM 22.26 (k) MemNet+SFM 22.13 (l) RIDNet+SFM 22.12

Figure 2.19 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 21 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).

(a) Noisy ($\sigma = 50$) (b) DnCNN [150] 23.51 (c) N2N [80] 23.04

(d) Ground-truth (e) DnCNN+SFM 23.87 (f) N2N+SFM 23.36

(g) N3Net [103] 23.40 (h) MemNet [125] 23.05 (i) RIDNet [6] 23.15

(j) N3Net+SFM 23.95 (k) MemNet+SFM 24.01 (l) RIDNet+SFM 23.84

Figure 2.20 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 23 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).

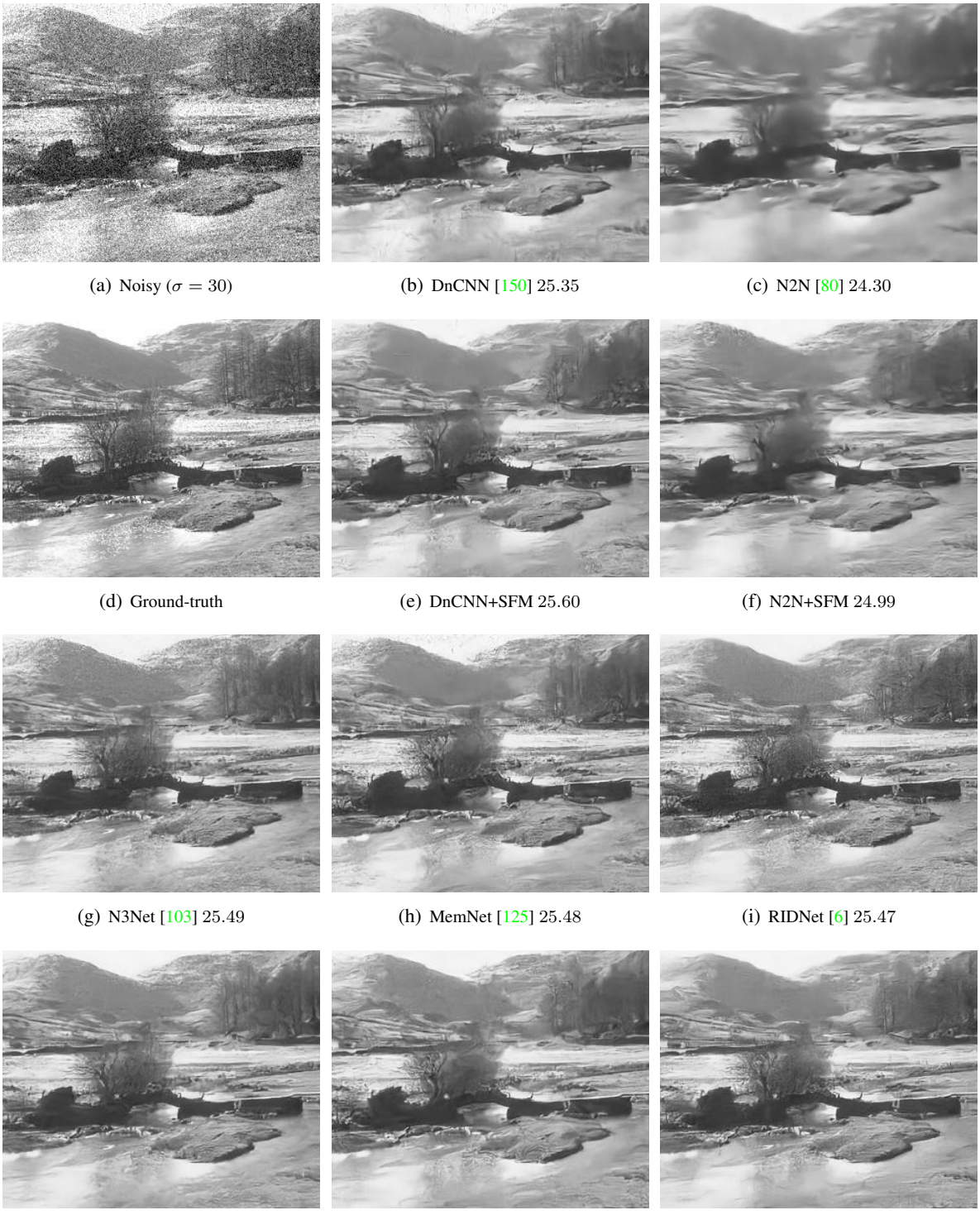
(a) Noisy ($\sigma = 50$) (b) DnCNN [150] 23.77 (c) N2N [80] 23.21

(d) Ground-truth (e) DnCNN+SFM 24.26 (f) N2N+SFM 23.83

(g) N3Net [103] 24.10 (h) MemNet [125] 23.58 (i) RIDNet [6] 23.45

(j) N3Net+SFM 24.30 (k) MemNet+SFM 24.31 (l) RIDNet+SFM 24.30

Figure 2.21 – Denoising results of different methods (1st and 3rd row), and of the same methods trained with our SFM (2nd and 4th row), for image 47 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).



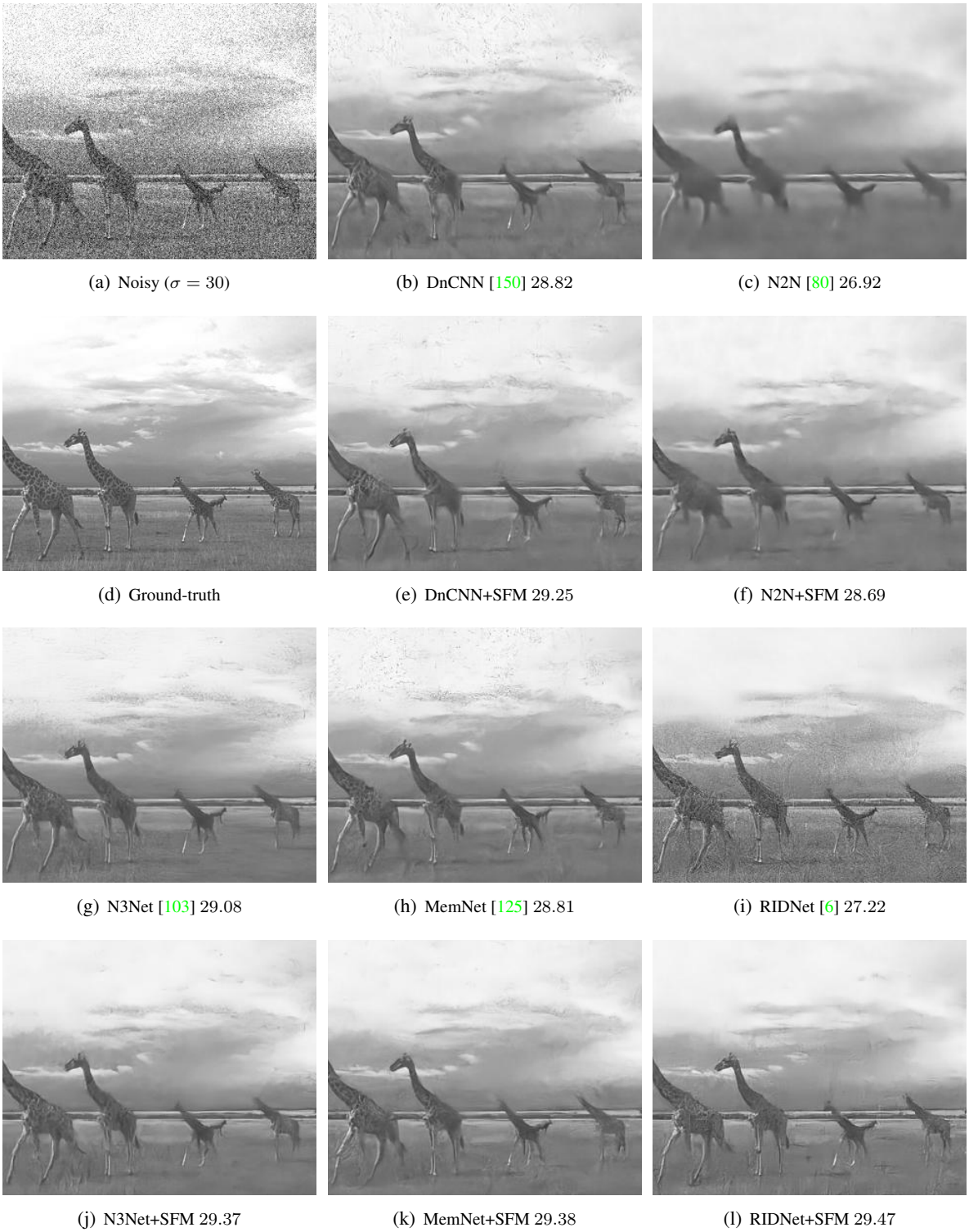
(a) Noisy ($\sigma = 30$) (b) DnCNN [150] 25.35 (c) N2N [80] 24.30

(d) Ground-truth (e) DnCNN+SFM 25.60 (f) N2N+SFM 24.99

(g) N3Net [103] 25.49 (h) MemNet [125] 25.48 (i) RIDNet [6] 25.47

(j) N3Net+SFM 25.58 (k) MemNet+SFM 25.63 (l) RIDNet+SFM 25.66

Figure 2.22 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 49 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).



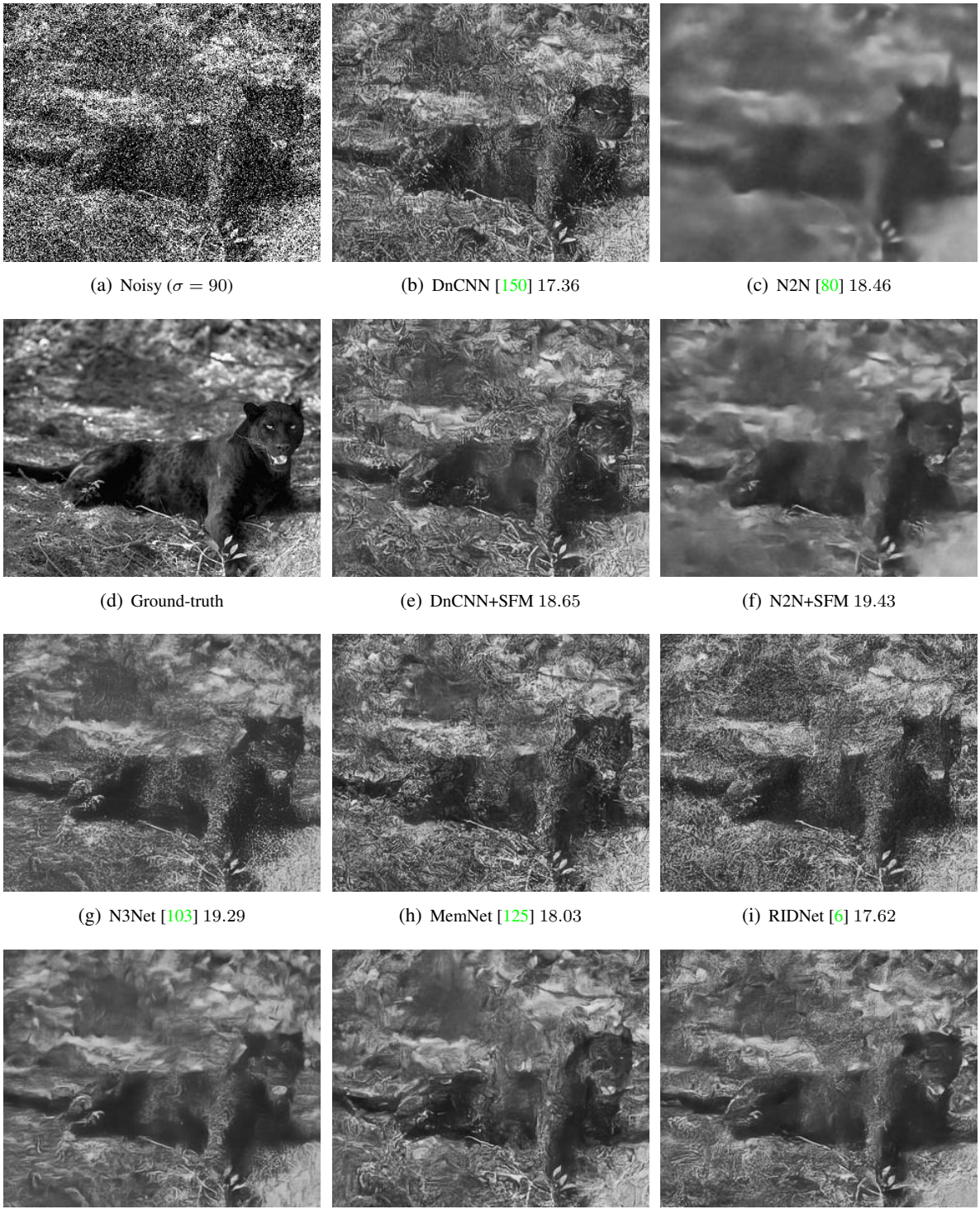
(a) Noisy ($\sigma = 30$) (b) DnCNN [150] 28.82 (c) N2N [80] 26.92

(d) Ground-truth (e) DnCNN+SFM 29.25 (f) N2N+SFM 28.69

(g) N3Net [103] 29.08 (h) MemNet [125] 28.81 (i) RIDNet [6] 27.22

(j) N3Net+SFM 29.37 (k) MemNet+SFM 29.38 (l) RIDNet+SFM 29.47

Figure 2.23 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 51 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).



(a) Noisy ($\sigma = 90$) (b) DnCNN [150] 17.36 (c) N2N [80] 18.46

(d) Ground-truth (e) DnCNN+SFM 18.65 (f) N2N+SFM 19.43

(g) N3Net [103] 19.29 (h) MemNet [125] 18.03 (i) RIDNet [6] 17.62

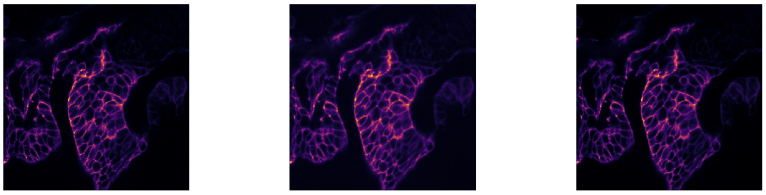
(j) N3Net+SFM 19.93 (k) MemNet+SFM 19.39 (l) RIDNet+SFM 19.44

Figure 2.24 – Denoising results with different methods (1st and 3rd row), and with the same method trained with our SFM (2nd and 4th row), for image 62 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).

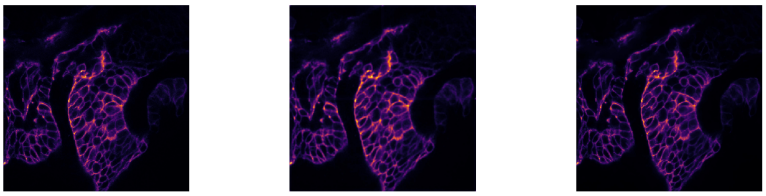
(a) Noisy ($\sigma = 80$) (b) DnCNN [150] 19.95 (c) N2N [80] 20.67 (d) N3Net [103] 21.45 (e) MemNet [125] 20.72 (f) RIDNet [6] 20.35

(g) Ground-truth (h) DnCNN+SFM 20.99 (i) N2N+SFM 21.14 (j) N3Net+SFM 21.89 (k) MemNet+SFM 21.72 (l) RIDNet+SFM 21.71

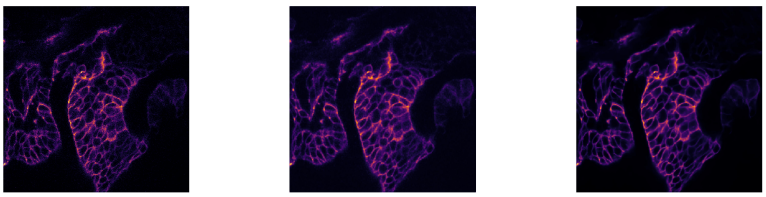
Figure 2.25 – Denoising results with different methods (top row), and with the same method trained with our SFM (bottom row), for image 63 of the BSD68 benchmark. We also show the PSNR values of every denoised result (in $dB$).



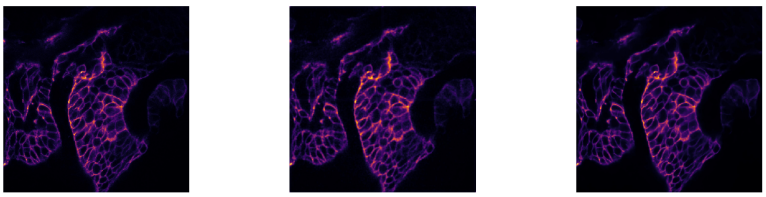
(a) Noisy (16 raw avg) (b) N2S [9] (c) N2N [80]



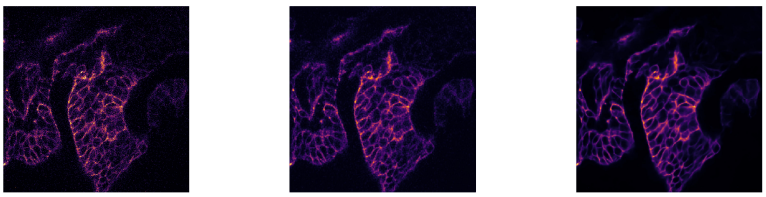
(d) GT (e) N2S + SFM (f) N2N + SFM



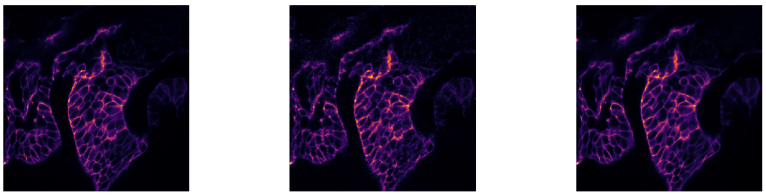
(g) Noisy (4 raw avg) (h) N2S [9] (i) N2N [80]



(j) GT (k) N2S + SFM (l) N2N + SFM

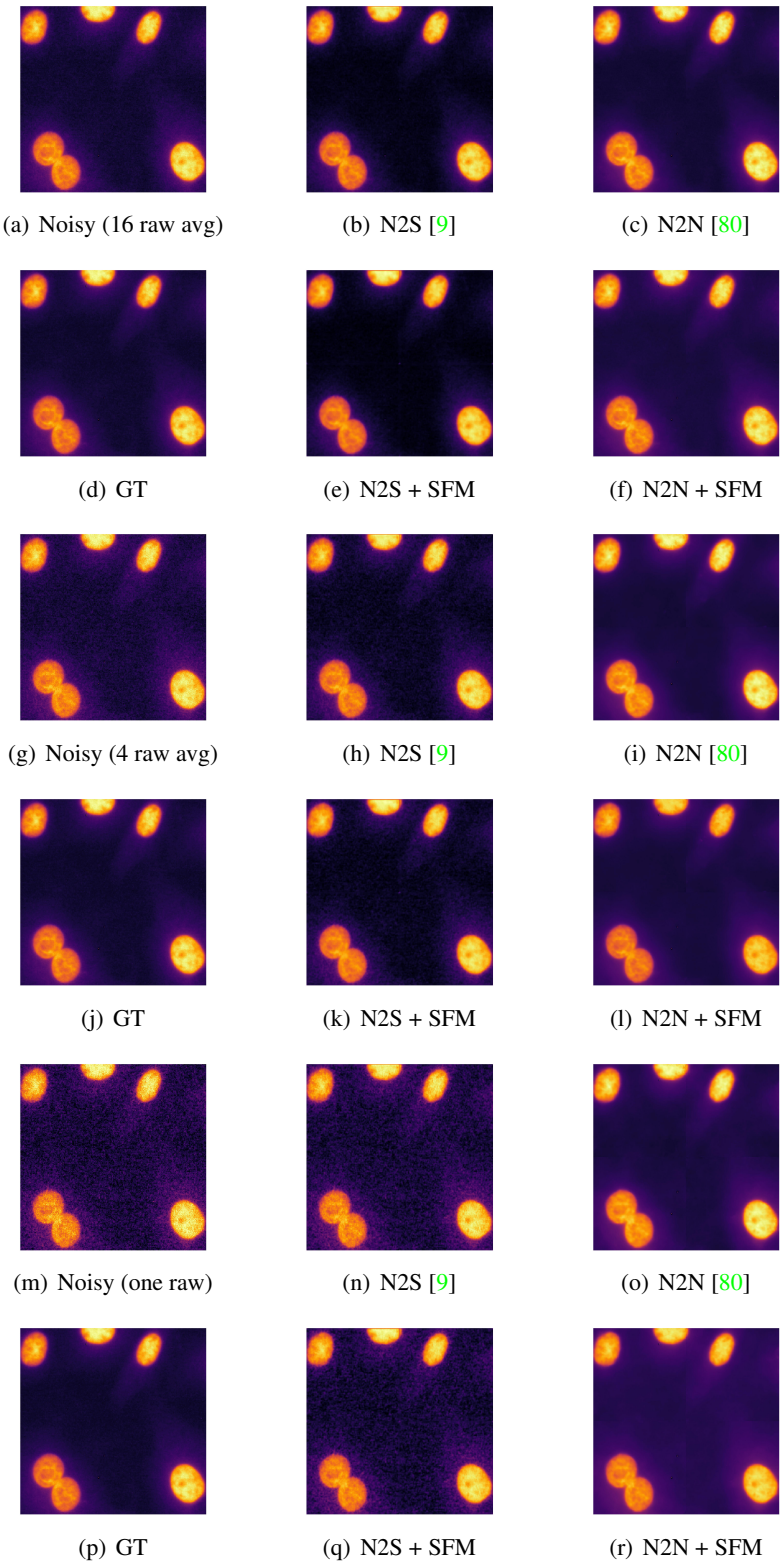


(m) Noisy (one raw) (n) N2S [9] (o) N2N [80]



(p) GT (q) N2S + SFM (r) N2N + SFM

Figure 2.26 – Confocal microscopy sample results for denoising the noisy input image from the real fluorescence microscopy denoising dataset [158]. The first image (a) averages 16 raw images to obtain the noisy input, the second one (g) averages 4 raw images, and the last one (m) is directly a raw image. The 'ground-truth' images are estimated by averaging 50 raw images [158].



(a) Noisy (16 raw avg) (b) N2S [9] (c) N2N [80]

(d) GT (e) N2S + SFM (f) N2N + SFM

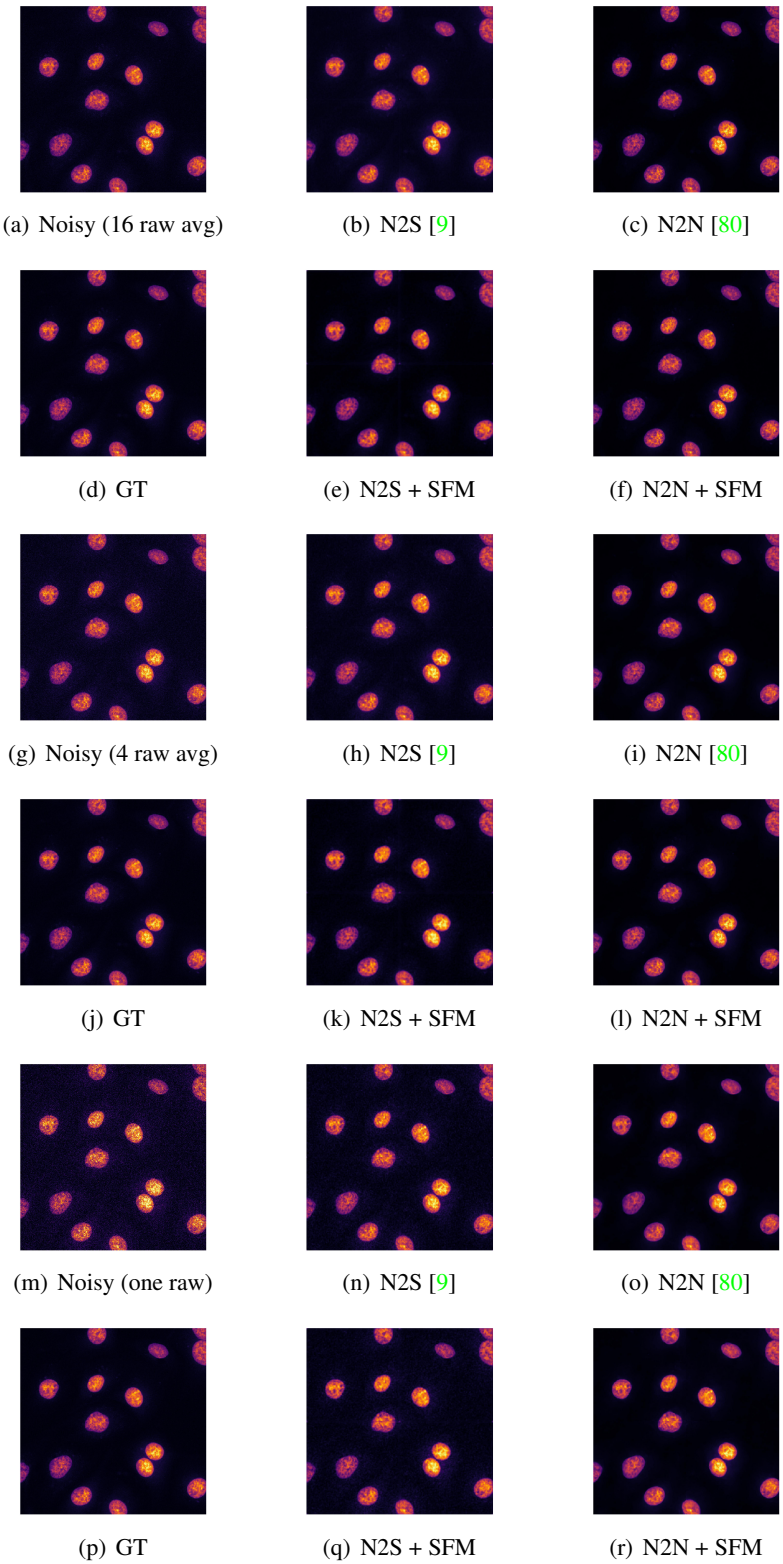
(g) Noisy (4 raw avg) (h) N2S [9] (i) N2N [80]

(j) GT (k) N2S + SFM (l) N2N + SFM

(m) Noisy (one raw) (n) N2S [9] (o) N2N [80]

(p) GT (q) N2S + SFM (r) N2N + SFM

Figure 2.27 – Widefield microscopy sample results for denoising the noisy input image from the real fluorescence microscopy denoising dataset [158]. The first image (a) averages 16 raw images to obtain the noisy input, the second one (g) averages 4 raw images, and the last one (m) is directly a raw image. The 'ground-truth' images are estimated by averaging 50 raw images [158].



(a) Noisy (16 raw avg) (b) N2S [9] (c) N2N [80]

(d) GT (e) N2S + SFM (f) N2N + SFM

(g) Noisy (4 raw avg) (h) N2S [9] (i) N2N [80]

(j) GT (k) N2S + SFM (l) N2N + SFM

(m) Noisy (one raw) (n) N2S [9] (o) N2N [80]

(p) GT (q) N2S + SFM (r) N2N + SFM

Figure 2.28 – Two-photon microscopy sample results for denoising the noisy input image from the real fluorescence microscopy denoising dataset [158]. The first image (a) averages 16 raw images to obtain the noisy input, the second one (g) averages 4 raw images, and the last one (m) is directly a raw image. The 'ground-truth' images are estimated by averaging 50 raw images [158].

# Chapter 3

# Exploring Bayesian Optimality in Deep Gaussian Image Denoising

In the previous chapter, we gained further insight into the actual data fidelity and prior learning of deep restoration networks. In this chapter, we investigate the optimality of network learning by synthetically defining our own theoretical prior model on a denoising, or generally a signal unmixing, problem. We also propose internal network modifications for guiding the network towards that statistically optimal learning.

Blind and universal image denoising uses a unique model that denoises images with any level of noise. It is especially practical as noise levels do not need to be known when the model is developed nor at test time. We propose a theoretically grounded blind and universal deep learning image denoiser for additive Gaussian noise removal. Our network is based on an optimal denoising solution that we call fusion denoising. It is derived theoretically with a Gaussian image prior assumption. Synthetic experiments show our network's generalization strength on unseen additive noise levels. We also adapt the fusion denoising network architecture for image denoising on real images. Our approach improves PSNR results of real-world grayscale additive image denoising on the training noise levels and on the noise levels not seen during training. It also improves state-of-the-art color image denoising performance on every single noise level, by an average of $0.1dB$, whether trained on or not.

Our code and models are made publicly available at https://github.com/majedelhelou/BUIFD
This work is published in the IEEE Transactions on Image Processing (TIP), vol. 29, 2020. [45]

## 3.1 Introduction

Aside from being a fundamental image restoration task, image denoising can also be part of deep network models in order to improve the training of high-level vision tasks [82] or can be used as a general regularizer [28]. However, as it is an ill-posed inverse problem, denoising is challenging [47]. After the development of the best analytical solution, BM3D [29, 58], little improvement in denoising performance had been achieved until the advent of deep-learning-based denoisers [150]. Recent convolutional neural-network-based methods achieve state-of-the-art image denoising performance and are even faster than traditional optimization-based approaches [143]. The increased capacity of deep CNN models also addresses the limitation of previous multi-layer perceptron methods when it comes to denoising different levels of noise [17]. Well-designed CNN architectures can also outperform adversarial training methods in image restoration tasks [121].

Neural networks can be deep and wide hence have a large capacity to model complex functions [147, 152], by using network regularization or normalization [62] and residual learning [59]. However, the complex functions modeled by the networks are not interpretable and have little connection to stochastic denoising. This is a limitation for training general models for denoising different noise levels. Denoisers are *blind* when they require no information about the noise level at test time, and *universal* when a single model can handle all noise levels. Blind universal models are important because knowing the noise level, at test time or ahead of training, is not a practical scenario for most applications.

We first mathematically derive a blind and universal denoising function under the theoretical assumption that the image prior is Gaussian. Our denoising function, which is optimal in stochastic expectation, is referred to as fusion denoising because it fuses the input with a prior weighted using the signal-to-noise ratio. It is optimized for additive Gaussian noise removal. Our experimental results show that the state-of-the-art denoiser DnCNN [150] can model an optimal fusion denoising function. However, it only models it for noise levels that are seen by the network during training. For unseen levels, our synthetic experiment's fusion network, called *Fusion Net*, far outperforms DnCNN. We show on synthetic data our improved generalization results.

The assumption that the image prior is Gaussian does not necessarily apply to real-world images. Building on the foundations of our theoretical solution, we adapt our *Fusion Net* by designing a second network that *learns* a fusion function for additive Gaussian noise removal. We call this new network *Blind Universal Image Fusion Denoiser (BUIFD)*. BUIFD improves state-of-the-art denoising performance on noise levels seen in training for grayscale and color images on the standard Berkeley test sets (BSD68 and CBSD68) [108]. Furthermore, we show that our generalization results on unseen noise levels obtained in our synthetic experiment extend to the denoising of the grayscale BSD68 test set. Indeed, the denoising performance on noise levels not seen by the network during training improves by multiple PSNR points. We present an extended denoising evaluation that covers other test datasets and other traditional and learning-based denoising methods.

Our main contributions in this chapter are as follows: (1) We theoretically derive an optimal fusion denoising function and integrate it into a deep learning architecture (Fusion Net) in order to evaluate the optimality of deep networks on a theoretical additive Gaussian noise removal task with a known prior. (2) We show, on synthetic data, that the integration of the auxiliary fusion loss into our Fusion Net improves the

network's generalization strength and brings it closer to the optimal solution. And (3) we develop a blind universal image fusion denoiser (BUIFD) network adapted to real images, and we show that it outperforms the state of the art for Gaussian noise removal on multiple standard image processing test sets.

This chapter is organized as follows. After a review of related work, we first lay the groundwork for our theoretical experiments. Our experiment enables us to assess the optimality of the networks on training noise levels and on the generalization of trained networks to unseen Gaussian noise levels, in comparison to the optimal Bayesian solution. We then extend the Bayesian-framework solution into our network designed for real images (BUIFD) whose exact prior is unknown to improve generalization. Experimental results on standard denoising benchmarks show that our denoising network outperforms the state of the art, especially on unseen noise levels.

## 3.2 Related Work

Image denoising approaches in the literature can be divided into classical methods and the more recent deep-learning-based methods. One common aspect is, however, the use of image priors for the improvement of denoising results. For practical reasons, it is important for a denoiser to be blind and universal as the noise levels in noisy images might not be constant or known.

**Image Priors.** Image priors are essential for denoising, whether they are in the form of assumptions made on image gradients [71, 95, 110, 136], sparsity [48, 36], self-similarity within images [37, 16, 139], hybrid approaches [88], or neural network weights given a certain architecture [150, 14]. Even traditional methods based on diffusion or filtering (in space [101] or in other domains [119]) rely on some priors. In all their forms and for multiple image restoration problems, they can be discovered and tested heuristically [71, 43], learned with dictionaries [48], learned with Markov random fields [108], or learned with deep neural networks [150]. In our network, the prior takes the explicit form of learned feature representations.

**Noise Modeling.** Additive white Gaussian noise is not necessarily the best model in practical scenarios such as denoising raw images [14]. Nevertheless, a large part of the image denoising literature focuses on Gaussian denoising as it remains a fundamental problem. Images with noise that follows different, potentially data-dependent, distributions can be transformed into images with Gaussian noise and can be transformed back [90, 102]. Furthermore, a Gaussian denoising solution can serve as a proximal [98, 79] for image regularizers. It can be a substitute for the costly step in half-quadratic splitting (HQS) optimization, typically responsible for non-differentiable regularization in image processing. This approach is taken in the recent HQS method that uses the denoiser for image restoration [151]. Therefore, we work with the assumption of an additive white Gaussian noise model.

We refer the reader to Section 1.2 for related work on image denoisers and, in the following section, we focus specifically on the blind and universal properties of image denoisers.

**Blind Universal Denoisers.** The state-of-the-art Gaussian denoiser DnCNN is both universal and blind [150]. It is a deep network that is jointly trained on patches with a randomly sampled noise level in order to generalize denoising to a range of noise levels. It has not yet been outperformed by other methods, whether blind or not [128, 53]. Only the recent FFDNet [152] by the same authors of DnCNN [150] improves on DnCNN for noise levels $50$ and $75$ by $0.06$ and $0.15dB$, respectively, on the Berkeley BSD68

set, whereas it performs similarly or worse for other levels. It is, however, not a blind network as it requires a noise-level map as input. Lefkimmiatis [79] recently studied universal denoising, by building on prior work for modeling patch similarity in CNNs [78]. His methods are, strictly speaking, not universal as two networks are trained separately, one for low ($\leq 30$) and one for high noise levels ($\in [30, 55]$). Hence, they are non-blind because a noise-level-based choice must be made at inference time. Furthermore, the published results do not outperform the blind DnCNN denoising results. Therefore, we conduct evaluation comparisons of our BUIFD method with the state-of-the-art DnCNN and the classic BM3D approach [29, 31], which is the best non-learning-based denoiser. It uses image self-similarities by jointly filtering similar image patches. The authors also present a blind version of the BM3D algorithm, and we compare it to both the blind and non-blind versions.

Our proposed image denoiser BUIFD learns to disentangle its features in order to predict a prior and a noise level intermediate results. They serve as inputs to the fusion part of the network, the part that is responsible for the final denoising. Disentangling the feature space is fundamental for interpretability [23], partial transfer learning [148], domain translation [142], domain adaptation [149], specific attribute manipulation [42, 83, 161] and multi-task networks [13]. In our case, it is fundamental for our theoretical denoising function as the different representations serve as its inputs.

## 3.3 Single-Image Fusion Denoising

In this section, we present a theoretically designed experiment that enables us to evaluate the optimality of a deep denoiser. We incorporate, based on our theoretical framework's optimal solution, a structural modification to the network, and we show the improved generalization strength of our novel architecture. We discuss the internal disentangled learning that takes place in this architecture and we propose a partial supervision on the intermediate feature space. We extend, to a deep network for denoising real images, this theoretical framework where we synthesized our *image* prior, and we discuss its relation with the Bayesian framework out of which we derived this more general solution.

### 3.3.1 Theoretical Framework

Although some specific applications can have a more accurate modeling [74, 133], an additive white Gaussian noise model is often assumed in denoising tasks, as it models common acquisition channels [137]. Hence, we assume that the additive independent and identically distributed noise $n$ follows a Gaussian distribution $\mathcal{N}(0, \sigma_n^2)$ and is uncorrelated with the data $x$. The noise standard deviation $\sigma_n$ is called the noise level. In a Bayesian framework, the conditional probability distribution of the noiseless data $x$ given a noisy observation $y$ (where $y = x + n$) is given by the relation

$$P_{X|Y}(x|y) = \frac{P_{Y,X}(y,x)}{P_Y(y)} = \frac{P_{Y|X}(y|x)P_X(x)}{P_Y(y)}, \tag{3.1}$$

where $X$ and $Y$ are the random variables corresponding, respectively, to $x$ and $y$. We are interested in the conditional distribution as we search for the Maximum A Posteriori (MAP) estimate $\hat{x}$ of $x$. The former is

$$\hat{x} = \arg\max_{x} P_{X|Y}(x|y). \tag{3.2}$$

We also model the data prior on $x$ as a Gaussian distribution $\mathcal{N}(\bar{x}, \sigma_x^2)$ centered at $\bar{x}$ [106]. We later modify this assumption in Section 3.3.4 to the practical case of real-world images. The conditional probability of $y$ given a noiseless $x$ value is

$$P_{Y|X}(y|x) = \frac{1}{\sqrt{2\pi\sigma_n^2}} e^{-\frac{(y-x)^2}{2\sigma_n^2}}, \tag{3.3}$$

and the probability distribution of $y$ is the convolution of those of $x$ and $n$, given in the Gaussian case by

$$P_Y(y) = P_X(x) \circledast P_N(n) = \frac{e^{-\frac{(y-\bar{x})^2}{2(\sigma_x^2+\sigma_n^2)}}}{\sqrt{2\pi(\sigma_x^2+\sigma_n^2)}}, \tag{3.4}$$

where $\circledast$ is the convolution operator. With these probability distribution functions, we can obtain an expression for the conditional distribution of $x$, given its noisy observation $y$ by substituting Equation (3.3) and Equation (3.4) into Equation (3.1)

$$P_{X|Y}(x|y) = \frac{e^{-\frac{(x-\bar{x})^2}{2\sigma_x^2} - \frac{(y-x)^2}{2\sigma_n^2} + \frac{(y-\bar{x})^2}{2(\sigma_x^2+\sigma_n^2)}}}{\sqrt{2\pi(\sigma_x^2\sigma_n^2)/(\sigma_x^2+\sigma_n^2)}}. \tag{3.5}$$

And $P_{X|Y}(x|y)$ can also be written in the following form of a Gaussian in $x$, given an observation $y$

$$P_{X|Y}(x|y) = \frac{1}{\sqrt{2\pi\hat{\sigma}_x^2}} e^{-\frac{(x-\hat{\mu})^2}{2\hat{\sigma}_x^2}}. \tag{3.6}$$

By matching the expanded expression of $P_{X|Y}(x|y)$ with Equation (3.6) for all possible $x$ values, we obtain the expressions for $\hat{\mu}$ and $\hat{\sigma}^2$

$$\hat{\mu} = \frac{\sigma_n^2\bar{x} + \sigma_x^2 y}{\sigma_x^2+\sigma_n^2}, \qquad \hat{\sigma}^2 = \frac{\sigma_x^2\sigma_n^2}{\sigma_x^2+\sigma_n^2}. \tag{3.7}$$

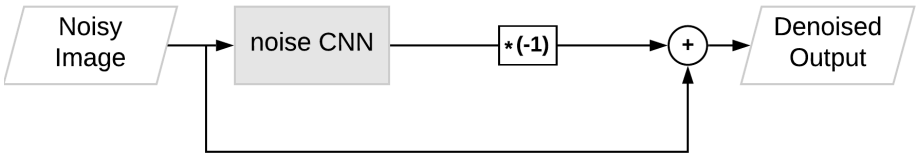
For the Gaussian shown in Equation (3.6), the MAP estimator is also the conditional expected value (mode and mean being equal), hence it is given by

$$\hat{x} = \mathbb{E}[x|y] = \int_{-\infty}^{\infty} x \cdot P_{X|Y}(x|y)dx, \tag{3.8}$$

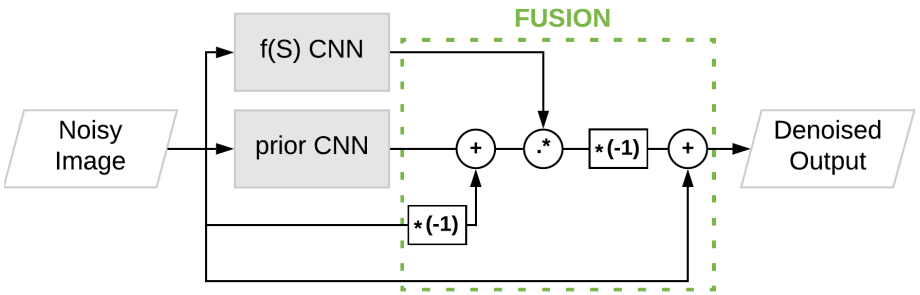
which, by using Equation (3.6), can be directly derived to be

$$\hat{x} = \frac{\bar{x}}{1+S} + \frac{y}{1+1/S}, \tag{3.9}$$

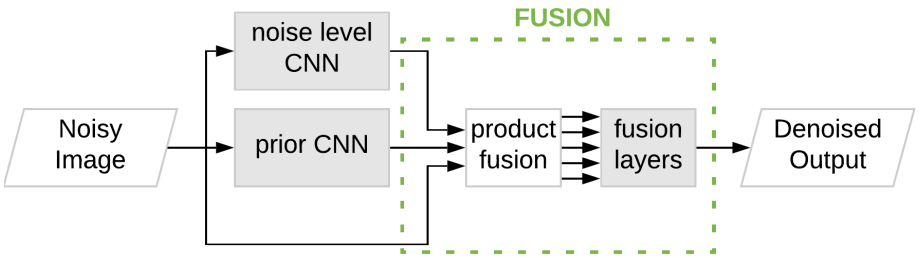
where $S \triangleq \sigma_x^2/\sigma_n^2$ and stands for signal-to-noise ratio. We call this operation fusion denoising as it fuses the prior and the noisy image, based on the SNR.




(a) DnCNN




(b) Fusion Net




(c) BUIFD

Figure 3.1 – (a) Schematic of the DnCNN residual-learning approach for denoising. The network predicts the noise in an image. (b) Our Fusion Net that explicitly learns the SNR function for optimal fusion of the noisy image with the learned prior, following Equation (3.9). (c) Our real-image fusion denoiser, BUIFD, where fusion is carried out with a pixel-wise product stage followed by three convolution layers for learning a general fusion function (Section 3.3.4).

Image denoising models are typically trained to maximize PSNR or equivalently minimize mean squared error (MSE) loss. This means that with close-to-optimal convergence of a neural network model (MSE loss $\rightarrow 0_+$), its output tends towards the minimum MSE estimator (MMSE). With our Gaussian modeling, this leads to the MAP estimator $\hat{x}$ of Equation (3.9). Hence, an MSE reconstruction loss in a neural network leads to the estimator $\hat{x}$, *iff* $S$ and $\bar{x}$ are correctly predicted and correctly used in the fusion with the noisy input $y$, as in Equation (3.9). The optimal fusion, used as reference in our experimental evaluation in Section 3.4.2, is given the exact $S$ and $\bar{x}$ values for Equation (3.9).

| | Blind training noise levels | | | | |
|---|---|---|---|---|---|
| Noise $\sigma$ | **5** | **10** | **15** | **20** | **25** |
| Optimal Fusion | 34.325 | 28.778 | 25.947 | 24.261 | 23.185 |
| DnCNN [150] | 34.158 | 28.736 | 25.920 | 24.245 | 23.169 |
| (Ours) Fusion Net | 34.158 | 28.734 | 25.922 | 24.249 | 23.173 |
| $p$-value | 0.760 | 0.568 | 0.465 | 0.100 | 0.053 |
| | Higher levels not seen during training | | | | |
| Noise $\sigma$ | **30** | **40** | **50** | **60** | **70** |
| Optimal Fusion | 22.464 | 21.604 | 21.138 | 20.860 | 20.681 |
| DnCNN [150] | 22.281 | 20.490 | 18.925 | 17.548 | 16.372 |
| (Ours) Fusion Net | **22.346** | **21.310** | **20.908** | **20.609** | **19.669** |
| $p$-value | $\approx 0$ | $\approx 0$ | $\approx 0$ | $\approx 0$ | $\approx 0$ |

Table 3.1 – Test set PSNR ($dB$) results for the noise standard deviations given in the top row. The networks are trained on noise levels randomly chosen in $[5, 25]$. Noise levels in the right half of the table are not seen during training. We also report the optimal Bayesian denoising (Optimal Fusion). The bottom row shows the independent two-sample T-test results between DnCNN and our Fusion Net. The two-tailed $p$-values validate the null hypothesis of equal average PSNR between DnCNN and the Fusion Net on training noise levels, with significance level $0.05$.

### 3.3.2 Fusion Net Architecture

We incorporate the basic structure of the optimal fusion solution into the architecture of a neural network, which we call *Fusion Net*. We build the main blocks of our Fusion Net, based on the blind DnCNN introduced in [150] and illustrated in Figure 3.1(a). In Figure 3.1, the noise-predicting CNN of DnCNN (Figure 3.1(a)), the prior-predicting CNN, and the one predicting $f(S)$ (where $f(S) \triangleq \frac{1}{1+S}$) in our Fusion Net (Figure 3.1(b)), all use the same DnCNN architecture design. The CNNs are all constituted of a sequence of convolution layers, rectified linear units (ReLU) [94] and batch normalization blocks [62]. Note that $f(S)$ is inversely proportional to the SNR and proportional to the noise level. It is the factor multiplying the prior in Equation (3.9). To summarize, the $f(S)$ CNN predicts $\frac{1}{1+S}$ where $S$ is the SNR of the input image (determined by the noise level and the image model used in our theoretical settings), and the prior CNN predicts $\bar{x}$ defined in Equation (3.8).

Unlike the DnCNN that predicts the *noise* values in the input noisy image then subtracts them from the noisy input to yield the final denoised output, our network learns optimal *fusion denoising* given by the function in Equation (3.9), as illustrated in Figure 3.1(b). The same depth and capacity of the DnCNN are retained to learn separately the image prior and the SNR function, $f(S)$, which is required for the weighted fusion of the prior and the noisy input image. Note that SNR learning also contains a form of prior knowledge, but of variance rather than of expectation. We subtract from the prior our noisy input image and multiply the result, pixel-wise, with the SNR function. This yields the noise prediction given a noisy input that we subtract from the latter to obtain the denoised output. This architecture is mathematically equivalent to Equation (3.9). However, the wiring of Figure 3.1(b) enables us to have a clear residual-learning connection and to keep the parallelism between the two aforementioned networks.

### 3.3.3 Fusion Net Feature Disentangling

To mimic the optimal fusion between an image prior and a noisy image based on the SNR, as in Equation (3.9), both the architecture and loss function are adapted. For the fusion, the network needs to predict the image prior $\bar{x}$ and $f(S)$ per pixel (Figure 3.1(b)). We obtain that, with close-to-zero MSE reconstruction loss of our Fusion Net, the ground-truth target and the network output are approximately equal

$$\bar{x} \cdot f(S) + y \cdot (1 - f(S)) \approx a \cdot b + y \cdot (1 - b), \forall y \in \mathcal{D}^{\mathcal{T}}, \tag{3.10}$$

where $a$ and $b$ are the outputs of intermediate layers in the Fusion Net, and $y$ is the noisy input. Specifically, $a$ is the output of the final layer of the prior CNN in Figure 3.1(b), and $b$ the output of the last layer of $f(S)$ in the same figure. After gradient descent convergence, when the MSE reconstruction loss is close to zero, we obtain the approximate equality of the left and right terms in Equation (3.10). We can view this equation as a first-degree polynomial in the variable $y$. As Equation (3.10) holds for all $y$ in the training dataset $\mathcal{D}^{\mathcal{T}}$, we can apply coefficient equating, where the coefficients are $\{a \cdot b, 1 - b\}$ and $\{\bar{x} \cdot f(S), (1 - f(S))\}$. We thus obtain the approximate equality between $a$ and $\bar{x}$ and between $b$ and $f(S)$. Hence, the network intermediate outputs $\{a, b\}$ are, respectively, equal to the prior and the SNR function $\{\bar{x}, f(S)\}$, with close-to-zero MSE reconstruction loss $\forall y \in \mathcal{D}^{\mathcal{T}}$. This extends to other $y$ outside the dataset assuming that the latter is sufficiently general. We can further incorporate optimal-denoising information in the Fusion Net, under the theoretical settings described in Section 3.3.1, through explicit SNR learning with a dedicated loss term. The fusion representations, i.e., the prior $\bar{x}$ and $f(S)$, are further enforced through a penalty term for predicting $f(S)$ in the loss function. The full loss function $\mathcal{L}_f$ of the Fusion Net is given by

$$\mathcal{L}_f = \alpha ||a \cdot b + y \cdot (1 - b) - x||_2^2 + (1 - \alpha)||b - f(S)||_2^2, \tag{3.11}$$

where $\alpha$ is a weight parameter, the first term is the MSE reconstruction loss similar to that of the DnCNN, and the second term is a reconstruction loss for $f(S)$. Following Equation (3.10), $a \cdot b + y \cdot (1 - b)$ is the denoised output of the Fusion Net.

As a result, the Fusion Net minimizes the reconstruction loss over the denoised image by learning to predict the image prior and the SNR function values separately. Unlike the DnCNN residual-learning network, which only uses ground-truth noise-free images during training, the Fusion Net also uses explicit SNR information.

### 3.3.4 Denoising Non-Gaussian Images

Here, our main objectives are (1) to design a *Blind Universal Image Fusion Denoiser (BUIFD)* for real images, by adapting the theoretical fusion strategy integrated in our Fusion Net, (2) to evaluate the denoising performance of BUIFD on training noise levels, and (3) to assess the generalization to unseen noise levels with real images.

As a real image cannot be modeled with a simple Gaussian prior, our image fusion denoising network used for real images (BUIFD), shown in Figure 3.1(c), is adapted from the theoretical Fusion Net, shown in Figure 3.1(b), by modifying the fusion part. We replace the optimal mathematical fusion by a product fusion

step, followed by trainable convolution layers. We use three convolution layers to learn the data-dependent fusion function. The optimal fusion function $F$ is to be applied on the noisy input image $y$, the prior prediction, and the noise-level prediction

$$\hat{x} = F(y,\ f_P(y, \theta_P),\ f_N(y, \theta_N)), \tag{3.12}$$

where the prior-predicting and noise-level-predicting network functions are, respectively, $f_P$ and $f_N$; their corresponding learned parameters are $\theta_P$ and $\theta_N$, and the denoised estimate is $\hat{x}$. Intuitively, the prior-predicting network ($f_P$) is used to predict the expected value of the unknown real-word distribution, out of which the intensity of a given pixel is sampled, for each pixel. The noise-level-predicting network ($f_N$) predicts the noise level, which is used to control the weighted average between a prior and an observation. When the noise level is low, the actual observation can be given more weight, and when the noise level is high, the current observation is less reliable and the fusion increasingly resorts to the use of the prior estimation.

The optimal fusion $F$ can be approximated by $\hat{F}$ modeled with three convolution layers. However, we expect $F$ to contain pixel-wise inter-input multiplications similar to those of Equation (3.9). As such pixel-wise multiplications cannot be replicated with convolutions, we pass two additional inputs into the convolution layers that model $\hat{F}$. These two additional inputs are given by

$$f_P(y, \theta_P) \odot f_N(y, \theta_N), \quad y \odot (1 - f_N(y, \theta_N)), \tag{3.13}$$

where $\odot$ is pixel-wise multiplication. They are concatenated with the inputs of $F$ given in Equation (3.12), yielding five different inputs that are sent to $\hat{F}$. The two additional inputs reduce the learning burden of the convolution layers and improve the denoising performance. Note that we normalize $f_N(\cdot, \cdot) \in [0, 1]$. We call this pixel-wise multiplication step and the concatenation of the additional inputs the *product fusion* (shown in the pipeline of Figure 3.1(c)). These two fusion steps, specifically the product fusion and the three convolution layers, form $\hat{F}$ and realize point (1) above. The BUIFD's optimization loss is given by

$$\mathcal{L}_f = ||\hat{F}(\mathbf{C}) - x||_2^2 + ||f_N(y, \theta_N) - N||_2^2, \tag{3.14}$$

where $\mathbf{C}$ is the concatenation of the inputs listed in Equation (3.12) and Equation (3.13), namely, $\{y$, $f_P(y, \theta_P)$, $f_N(y, \theta_N)$, $f_P(y, \theta_P) \odot f_N(y, \theta_N)$, $y \odot (1 - f_N(y, \theta_N))\}$, $x$ is the ground-truth original image, and $f_N(y, \theta_N)$ and $N$ are, respectively, the predicted and ground-truth noise level values, normalized to $[0, 1]$. We discuss the relation between BUIFD (Figure 3.1(c)) and our theoretical Bayesian network Fusion Net (Figure 3.1(b)) in detail in the following section.

### 3.3.5 Relation with the Bayesian Framework

The Fusion Net in Figure 3.1(b) explicitly models the relation with the Bayesian solution in the theoretical experiments. We discuss, in this section, the relation between BUIFD (Figure 3.1(c)) and the Bayesian solution Equation (3.9). We first note that a Gaussian prior does not perfectly model real images, hence we expect that the real-image BUIFD network (Figure 3.1(c)) deviates from the Fusion Net (Figure 3.1(b)), from which it is inspired, to adapt to real images. However, as addressed in Section 3.3.4, the relation between

BUIFD and the Bayesian framework is very pertinent.

First, the product fusion Equation 3.13 explicitly creates *the same components* as in the Bayesian equation Equation (3.9). This product fusion, based on SNR, weighs noisy input and learned prior as in the Bayesian fusion. The fusion layers are only three convolutional layers with no non-linearities, in order to ensure that mostly an additive fusion of our Bayesian terms takes place with local smoothing and that the relation with the Bayesian solution is preserved as much as possible.

Second, we do not predict an image prior in the sense of a pixel intensity probability distribution, but only the expected mean of that *unknown* distribution. In the literature, priors are often probability distributions of image gradients, but our definition is quite distinctive. *Our prior is, per pixel, the expected value of the distribution out of which the pixel's intensity was sampled*. Even with noise-free images, we cannot know exactly the distribution (nor its mean), per pixel, in order to assess how much this definition is still respected in the BUIFD network with real images. However, all other Bayesian components are consistent, as well as the empirical results. Our improvement of $3.30dB$ at the unseen noise level 70 in the theoretical experiment is paralleled by an improvement of about $3dB$ at noise level 75 in the real-image BSD68 experiment.

We hope our methodology motivates future work in analyzing deep network optimality on theoretical experiments that are designed such that an optimal solution is known, and that it motivates deep network design inspired from Bayesian solutions.

## 3.4 Experiments

### 3.4.1 Fusion Net Experimental Setup

The networks are trained (and tested) with data generated synthetically according to the theoretical assumption of a Gaussian image prior, as defined in Section 3.3.1. The training data is composed of over 200k patches of size $40 \times 40$ pixels. Image pixel intensities for the training data are drawn at random from $\mathcal{N}(127, 25^2)$, following the Gaussian image prior assumption, and all values are normalized to $[0, 1]$ before the training through division by $255$ and clipping of all values outside the interval to the interval's closer bound when noise is added. For the testing data, 256 images of size $256 \times 256$ pixels are used, and they are created with the same procedure as that of the training data.

We train the networks for 50 epochs with mini-batches of size 128. We use the Adam optimizer [70] with an initial learning rate of $0.001$ that is decayed by a factor of $10$ every 30 epochs; the remaining parameters are set to the default values. The weight $\alpha$ in Equation (3.11) is set to $0.1$. We train the networks with multiple levels of noise. The standard deviation of the additive Gaussian noise is chosen uniformly at random within the interval $[5, 25]$ during the training. At the end of every epoch, the noise components are re-sampled, following the same procedure, but not the ground-truth images. For the testing phase, the networks are evaluated on test images where the added noise is also Gaussian, with a given standard deviation.

### 3.4.2 Fusion Net Evaluation

PSNR results of DnCNN, our Fusion Net, as well as the optimal upper bound, are presented in Table 3.1. The optimal upper-bound denoising performance is that of the optimal mathematical solution in Equation (3.9). We can see that both the DnCNN and the Fusion Net perform similarly on the training noise levels (left half of the table) and are very close to optimal. To validate that the results are indeed statistically similar, we analyze the distribution of PSNR values across the test set. A two-sided T-test (independent two-sample T-test) is used to evaluate the null hypothesis that the PSNR results of both networks have similar expected values. This test is chosen as we have the exact same sample sizes defined by the test dataset, and the variances of PSNR results are very similar. The T-test results are given in the bottom row of Table 3.1; and the null hypothesis holds for all configurations in the left half of the table (for a $0.05$ significance level, i.e., a $p$-value $\geq 0.05$). This shows that the Fusion Net, despite the modeling that mimics optimal denoising fusion and the additional training information to learn SNR values, performs similarly to the DnCNN. Therefore, DNCNN has enough capacity and learns optimal denoising. This, however, only holds for the noise levels seen during training by the networks, shown in the left half of Table 3.1. The confidence in the null hypothesis decreases with increasing test noise levels. With a significance level above $0.053$, the null hypothesis would be rejected even for noise level $25$.

The evaluation results on noise levels larger than $25$, which are not trained on by any of the networks, are reported in the right half of Table 3.1. For these larger noise levels, the null hypothesis is very clearly rejected because there is a growing performance gap between DnCNN and our Fusion Net. As variances are very small in our results, the $p$-value quickly drops to zero when there is a PSNR gap. The Fusion Net generalizes better to unseen noise levels, even performing close to optimal up to noise level $60$. The further we increase the noise level, the larger the performance gap becomes between the Fusion Net and the DnCNN. Although both networks perform well for the training noise levels, the Fusion Net learns a more general model and clearly outperforms on unseen noise levels.

### 3.4.3 Real-Image Experimental Setup

We use the referenced implementation by the authors of DnCNN and the same datasets. As mentioned in Section 3.3.4, the architecture of our prior-predicting network is identical to that of DnCNN. All the network details are available in [150], and we omit the repetition. The same network depth and feature layers are used in the prior-predicting network (18 main blocks) in Figure 3.1(c). The noise-level network is a more shallow one that consists of five blocks similar to those used in the prior predictor. Each block is a convolution followed by a batch normalization and a ReLU, and we append to the noise-level predictor a convolution followed by an application of the logistic sigmoid function to obtain the normalized $f_N(\cdot, \cdot) \in [0, 1]$. The noise level values are mapped during the training to the range $[0, 1]$ by dividing by the largest *training* noise level. The three convolution layers approximating the final fusion have 16 channels. Both the BUIFD and the DnCNN networks are trained with the same training parameters and optimization settings, similar to Section 3.4.1 except for the patch size. For completeness, we provide all the details of the training hyper-parameters. We use the Adam optimizer [70] with an initial learning rate of $0.001$ that is decayed by a factor of $10$ every $30$ epochs, the remaining optimizer parameters being set to the default values. The networks are

https://github.com/SaoYan/DnCNN-PyTorch

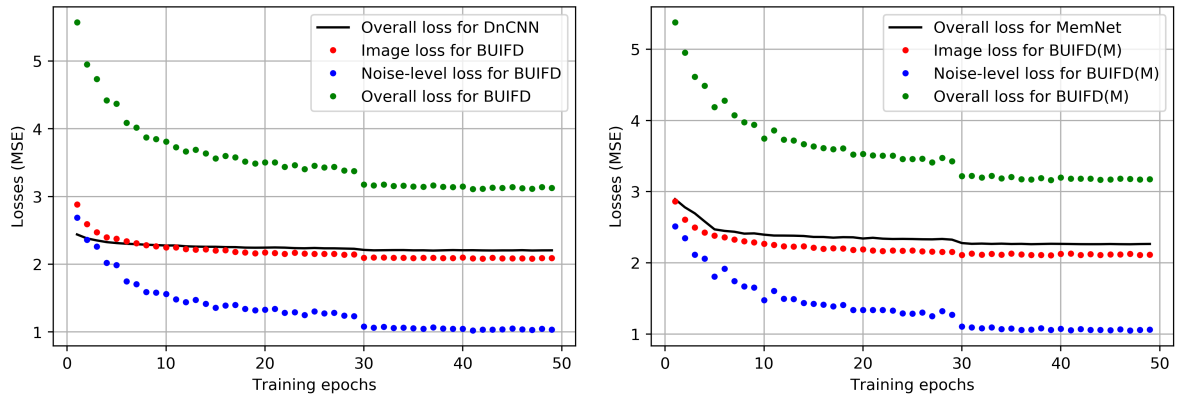
| | | Test noise level (standard deviation) | | | |
|---|---|---|---|---|---|
| **Method** | **Blind** | **5** | **10** | **15** | **20** |
| BM3D [29] | **No** | 37.57/0.964 | 33.27/0.916 | 30.98/0.871 | 29.45/0.831 |
| | Yes | 29.34/0.806 | 29.18/0.802 | 28.95/0.799 | 28.69/0.798 |
| $\text{DnCNN}_{55}$ [150] | Yes | **37.70/0.967** | **33.61**/0.926 | 31.31/0.882 | 29.65/0.838 |
| $\text{BUIFD}_{55}$ | Yes | 37.49/0.966 | 33.58/**0.926** | **31.40/0.888** | **29.91/0.852** |
| $\text{DnCNN}_{75}$ [150] | Yes | **37.64/0.967** | **33.62/0.927** | **31.37**/0.886 | 29.79/0.844 |
| $\text{BUIFD}_{75}$ | Yes | 37.25/0.964 | 33.47/0.924 | 31.35/**0.886** | **29.88/0.851** |
| | | **25** | **30** | **35** | **40** |
| BM3D [29] | **No** | 28.32/0.797 | 27.42/0.766 | 26.66/0.739 | 25.98/0.714 |
| | Yes | 28.32/0.797 | 27.32/0.762 | 25.13/0.638 | 22.39/0.494 |
| $\text{DnCNN}_{55}$ [150] | Yes | 28.31/0.795 | 27.17/0.754 | 26.19/0.717 | 25.31/0.682 |
| $\text{BUIFD}_{55}$ | Yes | **28.75/0.819** | **27.80/0.787** | **27.00/0.758** | **26.30/0.731** |
| $\text{DnCNN}_{75}$ [150] | Yes | 28.55/0.804 | 27.52/0.768 | 26.65/0.736 | 25.84/0.704 |
| $\text{BUIFD}_{75}$ | Yes | **28.74/0.819** | **27.82/0.788** | **27.01/0.759** | **26.32/0.732** |
| | | **45** | **50** | **55** | **60** |
| BM3D [29] | **No** | 25.28/0.686 | 24.79/0.667 | 24.30/0.648 | 23.86/0.632 |
| | Yes | 20.01/0.389 | 18.22/0.317 | 16.83/0.262 | 15.78/0.222 |
| $\text{DnCNN}_{55}$ [150] | Yes | 24.50/0.648 | 23.75/0.616 | 23.07/0.586 | 22.29/0.546 |
| $\text{BUIFD}_{55}$ | Yes | **25.65/0.704** | **25.06/0.680** | **24.52/0.658** | **23.97/0.637** |
| $\text{DnCNN}_{75}$ [150] | Yes | 25.14/0.675 | 24.48/0.647 | 23.90/0.621 | 23.34/0.597 |
| $\text{BUIFD}_{75}$ | Yes | **25.68/0.706** | **25.11/0.682** | **24.55/0.658** | **24.03/0.636** |
| | | **65** | **70** | **75** | **Mean** |
| BM3D [29] | **No** | 23.43/0.618 | 23.02/0.603 | 22.67/0.591 | 27.13/0.74 |
| | Yes | 14.86/0.189 | 14.10/0.165 | 13.48/0.147 | 22.17/0.51 |
| $\text{DnCNN}_{55}$ [150] | Yes | 21.06/0.460 | 19.42/0.352 | 17.88/0.278 | 26.08/0.67 |
| $\text{BUIFD}_{55}$ | Yes | **23.31/0.603** | **22.28/0.536** | **20.97/0.451** | **27.20/0.73** |
| $\text{DnCNN}_{75}$ [150] | Yes | 22.87/0.577 | 22.41/0.558 | 22.01/0.541 | 27.01/0.72 |
| $\text{BUIFD}_{75}$ | Yes | **23.56/0.617** | **23.10/0.598** | **22.66/0.582** | **27.37/0.75** |

Table 3.2 – PSNR ($dB$)/SSIM comparisons of *grayscale* image denoising on the BSD68 standard test set. We compare the non-blind BM3D, the blind BM3D, DnCNN, and our BUIFD. $\text{DnCNN}_{\sigma}$ or $\text{BUIFD}_{\sigma}$ indicates that the network sees noise levels *only* up to $\sigma$ during the training. Bold indicates the best blind result, for each range of training noise levels, and that best result is selected before rounding. Note that small deviations in reported PSNR values compared with the literature, notably on higher noise levels, are due to clipping noisy inputs, as a practical consideration.

trained for 50 epochs each, and the progress of the different losses can be seen in Figure 3.2. We use a patch size of $50 \times 50$ with a stride of 10 on the training images. The training mini-batch size is set to 128 patches per mini-batch. The added noise is drawn from a Gaussian distribution of given standard deviation based on the noise level. This standard deviation is sampled uniformly at random from a specified range (details in Section 3.4.4), and it is the same for all pixels in a given training patch. We use the training hyper-parameters of DnCNN to train it and to train BUIFD; the hyper-parameters are not tweaked for BUIFD. The noise-level predictor is jointly trained within BUIFD, hence both network branches always see the same training data (with the same simulated noise distributions) as each other in the experiments of Sec 3.4.4. We use the 400

| $\sigma_c$ | **15** | **25** | **40** | **55** | **65** |
|---|---|---|---|---|---|
| Non-blind BM3D | 29.30 | 27.80 | 25.75 | 24.28 | 23.41 |
| BM3D | 28.94 | 27.80 | 21.63 | 16.78 | 14.85 |
| $\text{DnCNN}_{55}$ | 31.24 | 28.32 | 25.41 | 23.17 | 20.83 |
| **$\text{BUIFD}_{55}$** | **31.38** | **28.74** | **26.22** | **24.33** | **22.81** |
| $\text{DnCNN}_{75}$ | 31.31 | 28.51 | 25.80 | 23.87 | 22.83 |
| **$\text{BUIFD}_{75}$** | **31.34** | **28.73** | **26.29** | **24.52** | **23.53** |

Table 3.3 – We evaluate PSNR values, with spatially varying noise level, on the BSD68 test set. The noise level increases linearly within the image over the range $[\sigma_c - 10, \sigma_c + 10]$. The non-blind BM3D is given the central noise level $\sigma_c$.




(a) DnCNN and BUIFD

(b) MemNet and BUIFD(M)

Figure 3.2 – Training losses of the different learning-based methods. Per epoch, we plot with a full black curve the overall loss (i.e., reconstruction loss) of the base methods DnCNN and MemNet, in (a) and (b) respectively. The same reconstruction loss with our fusion method is plotted with a dotted red curve, the noise-level loss computed on the corresponding intermediate output (i.e., the output of the noise level CNN) is plotted with a dotted blue curve, and the overall loss for the fusion methods (the sum of the former two losses) is plotted with a dotted green curve. Note the abrupt small improvement in loss reduction at epoch 30; this is when the learning rate is exponentially decayed. We can see that the different learned function converge by the end of training (logs shown for the methods with upper training noise level 55).

Berkeley images [24, 112] for grayscale training and the 432 color Berkeley images for color training, as in [150]. The same architectures are retained for grayscale and color networks.

### 3.4.4 Real-Image Evaluation

Grayscale denoising evaluation is made over the standard Berkeley 68 image test set (BSD68) [108] taken from [91]. Table 3.2 reports the results of our fusion approach and of the state-of-the-art blind DnCNN, when they are both trained with noise levels up to 55 or up to 75. Note that for our fusion approach that is trained up to noise level 55, we map the maximum network prediction of 1, during training, to 55 and not to

(a) Original $x$ (b) Noisy $y$ (level 25) (c) $f_P(y, \theta_P)$

(d) $f_N(y, \theta_N)$ (e) (Ours) **BUIFD**$_{55}$ $25.43dB$ (0.84 SSIM) (f) DnCNN$_{55}$ $24.23dB$ (0.66 SSIM)

(g) Original $x$ (h) Noisy $y$ (level 75) (i) $f_P(y, \theta_P)$

(j) $f_N(y, \theta_N)$ (k) (Ours) **BUIFD**$_{55}$ $20.18dB$ (0.55 SSIM) (l) DnCNN$_{55}$ $17.22dB$ (0.36 SSIM)

Figure 3.3 – Left to right: original and noisy images, prior and noise-level predictions of BUIFD, our fused denoising result and the DnCNN denoised image. Our denoising result is created by fusing the noisy image, the prior, and the noise level values, for instance (e) is $\hat{F}((b), (c), (d))$. All the networks are trained on noise levels in $[0, 55]$. Whether the noise level is seen (25), or not seen (75), during training, our denoised results show better noise removal: sky in (e-f), window, wall and arms in (k-l). We show the PSNR in $dB$ and the SSIM [135] between parentheses for the different results. Best viewed on screen.

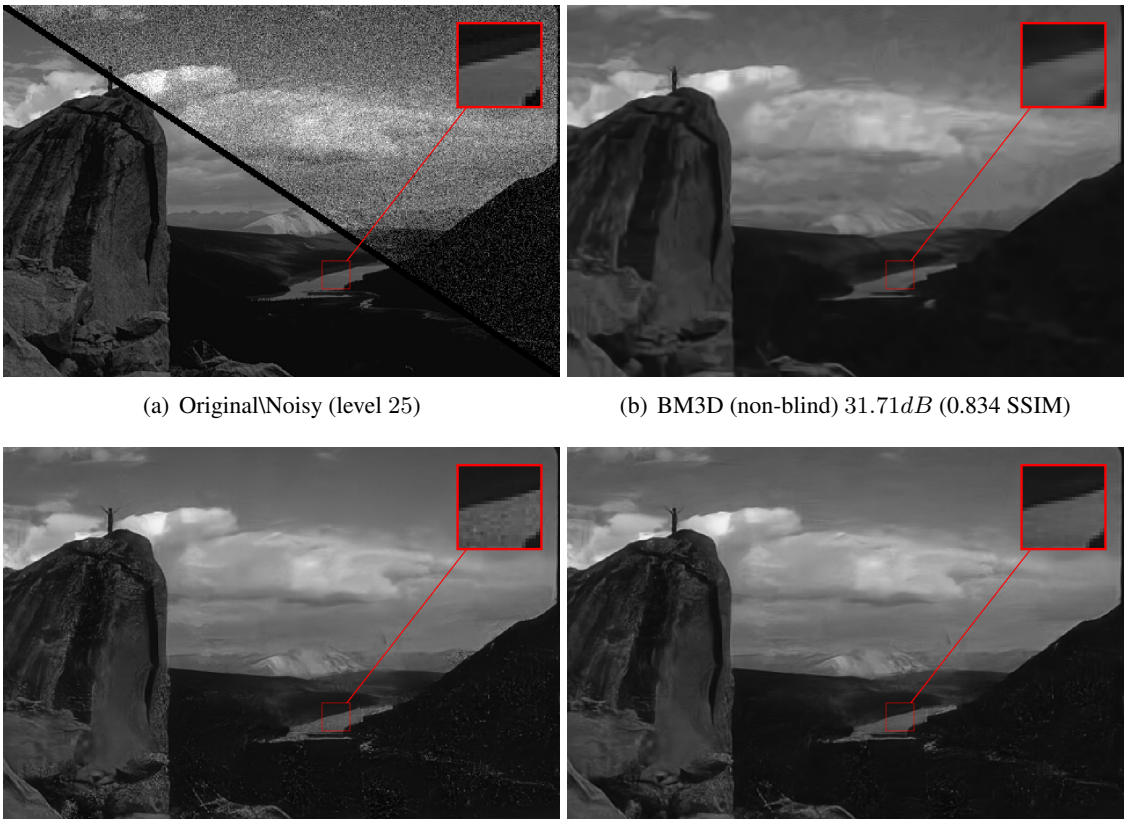
the maximum test noise level, for a more fair comparison. The results of the blind version of BM3D, as well

| | | Test noise level (standard deviation) | | | |
|---|---|---|---|---|---|
| **Method** | **Blind** | **5** | **10** | **15** | **20** |
| CBM3D [31] | **No** | 40.19/0.979 | 35.75/0.950 | 33.26/0.919 | 31.52/0.888 |
| | Yes | 28.17/0.772 | 28.08/0.769 | 27.94/0.765 | 27.74/0.760 |
| $\text{CDnCNN}_{55}$ [150] | Yes | 40.05/0.979 | 35.92/0.953 | 33.57/0.927 | 31.93/0.902 |
| $\text{CBUIFD}_{55}$ | Yes | **40.07/0.979** | **36.01/0.955** | **33.66/0.930** | **32.02/0.905** |
| $\text{CDnCNN}_{75}$ [150] | Yes | 39.75/0.978 | 35.74/0.953 | 33.46/0.928 | 31.86/0.903 |
| $\text{CBUIFD}_{75}$ | Yes | **40.05/0.980** | **35.98/0.955** | **33.65/0.930** | **32.03/0.906** |
| | | **25** | **30** | **35** | **40** |
| CBM3D [31] | **No** | 30.18/0.859 | 29.07/0.830 | 28.09/0.801 | 27.18/0.771 |
| | Yes | 27.49/0.754 | 27.21/0.748 | 26.90/0.743 | 26.58/0.738 |
| $\text{CDnCNN}_{55}$ [150] | Yes | 30.66/0.877 | 29.61/0.853 | 28.71/0.830 | 27.92/0.808 |
| $\text{CBUIFD}_{55}$ | Yes | **30.75/0.881** | **29.72/0.858** | **28.81/0.835** | **28.01/0.813** |
| $\text{CDnCNN}_{75}$ [150] | Yes | 30.61/0.879 | 29.59/0.855 | 28.70/0.833 | 27.92/0.812 |
| $\text{CBUIFD}_{75}$ | Yes | **30.76/0.883** | **29.71/0.860** | **28.81/0.838** | **28.01/0.816** |
| | | **45** | **50** | **55** | **60** |
| CBM3D [31] | **No** | 26.53/0.751 | 25.85/0.729 | 25.21/0.708 | 24.62/0.689 |
| | Yes | 26.23/0.733 | 25.85/0.729 | 25.41/0.720 | 24.83/0.695 |
| $\text{CDnCNN}_{55}$ [150] | Yes | 27.16/0.786 | 26.49/0.766 | 25.84/0.747 | 25.23/0.729 |
| $\text{CBUIFD}_{55}$ | Yes | **27.27/0.793** | **26.59/0.773** | **25.94/0.754** | **25.33/0.737** |
| $\text{CDnCNN}_{75}$ [150] | Yes | 27.19/0.792 | 26.52/0.772 | 25.89/0.753 | 25.27/0.735 |
| $\text{CBUIFD}_{75}$ | Yes | **27.28/0.796** | **26.60/0.776** | **25.96/0.758** | **25.34/0.740** |
| | | **65** | **70** | **75** | **Mean** |
| CBM3D [31] | **No** | 24.05/0.670 | 23.51/0.653 | 22.99/0.637 | 28.53/0.79 |
| | Yes | 24.05/0.647 | 23.07/0.581 | 21.93/0.508 | 26.10/0.71 |
| $\text{CDnCNN}_{55}$ [150] | Yes | 24.65/0.713 | 24.09/0.697 | 23.52/0.677 | 29.02/0.82 |
| $\text{CBUIFD}_{55}$ | Yes | **24.75/0.720** | **24.18/0.703** | **23.62/0.684** | **29.11/0.82** |
| $\text{CDnCNN}_{75}$ [150] | Yes | 24.69/0.717 | 24.13/0.701 | 23.59/0.684 | 28.99/0.82 |
| $\text{CBUIFD}_{75}$ | Yes | **24.76/0.722** | **24.18/0.705** | **23.64/0.689** | **29.12/0.82** |

Table 3.4 – PSNR ($dB$)/SSIM comparisons of *color* image denoising, similar to Table 3.2, on the CBSD68 standard test set. Bold indicates the best blind result, for each range of training noise levels, and that best result is selected before rounding.

as those of the non-blind BM3D that is given the correct test noise level at inference time, are also reported for reference. We restrict all noisy test images to the range $[0, 255]$, as having negative intensities, or values exceeding $255$, is not a configuration encountered in practice.

Figure 3.3 shows our intermediate feature results, the prior, and the noise level values, along with denoising results. The denoised image is created by fusing the noisy input image with the network-derived prior and the noise level values. The fusion is implemented by the product fusion step and the three convolution layers. As in practical scenarios, the denoised outputs are clipped to $[0, 255]$, as are the noisy input images. Our results remove, over low-frequency regions, the noise better than those of DnCNN, and the details are better reconstructed over the high-frequency content. We note that, at high noise levels, there is a smudging effect most visible around low-frequency regions (Figure 3.3 (k) and (l)), which creates blurry and noisy edges. These are created by both networks, but are more salient in our result (k) as it is less noisy



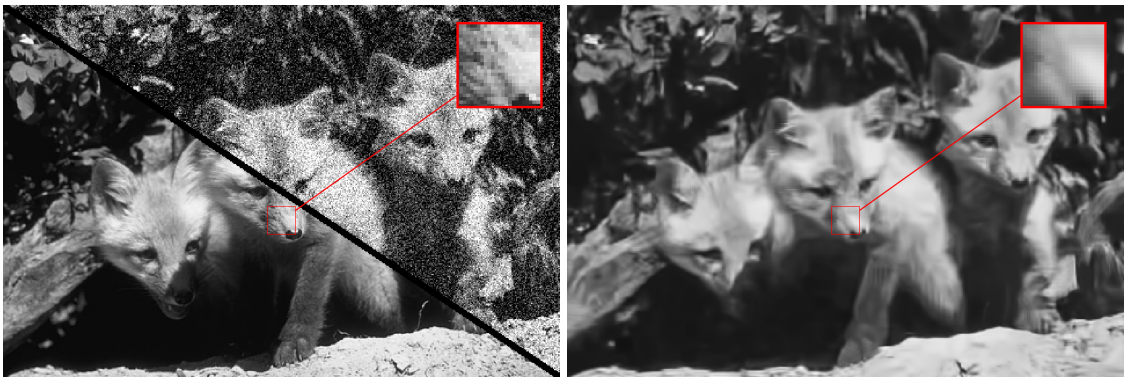
(a) Original\Noisy (level 25)

(b) BM3D (non-blind) $31.71dB$ (0.834 SSIM)

(c) DnCNN$_{75}$ $31.25dB$ (0.792 SSIM)

(d) (Ours) **BUIFD**$_{75}$ $31.81dB$ (0.841 SSIM)

Figure 3.4 – Grayscale image denoising example from BSD68. All networks are trained on all noise levels $[0, 75]$, and we test on noise level 25. Non-blind BM3D loses edge details due to blur smoothing. The network results are sharper, with the better PSNR being that of BUIFD$_{75}$. Best viewed on screen.



(a) Original\Noisy (level 45)

(b) BM3D (non-blind) $24.31dB$ (0.675 SSIM)

(c) DnCNN$_{75}$ $23.67dB$ (0.618 SSIM)

(d) (Ours) **BUIFD**$_{75}$ $24.43dB$ (0.677 SSIM)

Figure 3.5 – Grayscale image denoising example from BSD68. All networks are trained on all noise levels $[0, 75]$ and we test on noise level 45. Non-blind BM3D results are very smoothed, and details are lost. DnCNN preserves more details, but at the expense of PSNR. Our blind approach preserves details and outperforms the non-blind BM3D in terms of PSNR. Best viewed on screen.

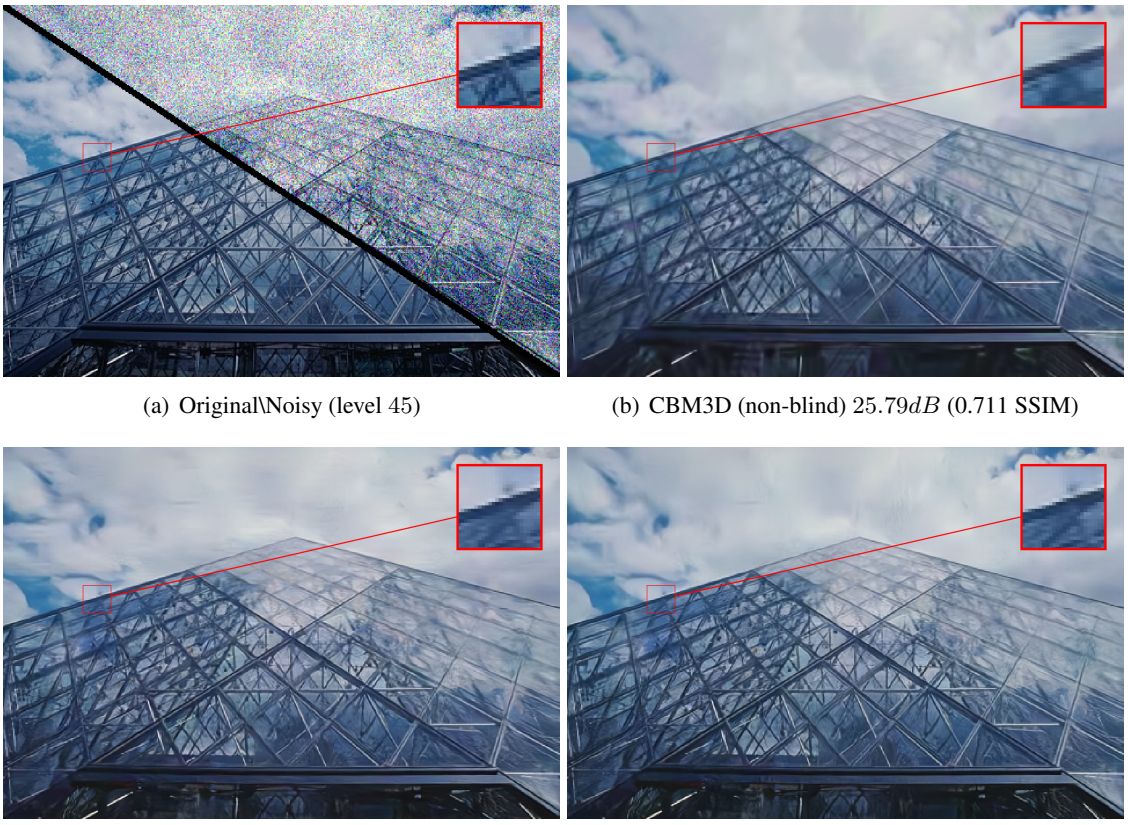
(a) Original\Noisy (level 25)

(b) CBM3D (non-blind) $29.81dB$ (0.852 SSIM)

(c) $\text{CDnCNN}_{75}$ $30.44dB$ (0.878 SSIM)

(d) (Ours) $\textbf{CBUIFD}_{75}$ $30.62dB$ (0.880 SSIM)

Figure 3.6 – Color image denoising example from CBSD68. All networks are trained on the full range of noise levels $[0, 75]$, and we test on noise level 25. Best viewed on screen.



(a) Original\Noisy (level 45)

(b) CBM3D (non-blind) $25.79dB$ (0.711 SSIM)

(c) CDnCNN$_{75}$ $26.43dB$ (0.764 SSIM)

(d) (Ours) **CBUIFD**$_{75}$ $26.68dB$ (0.775 SSIM)

Figure 3.7 – Color image denoising example from CBSD68. All networks are trained on the full range of noise levels $[0, 75]$, and we test on noise level $45$. Best viewed on screen.

than (l). The higher the noise level and standard deviation of the Gaussian noise are, the larger the number of averaged samples needs to be such that the statistical mean converges to zero. This makes the local mean of the noise across small patches vary randomly around zero from region to region and causes the smudging-like or wave-like effect (notice over low-frequency regions how almost all these artifacts have a curve shape, rather than a linear one that is modeled by the various different mean values around them).

As seen in Table 3.2, our fusion approach improves the PSNR at every single noise level, starting from $15 - 20$, which includes seen levels for both training ranges. Comparing $\text{DnCNN}_{75}$ and $\text{BUIFD}_{75}$, which are trained on all noise levels, we also note, with our approach, an improvement of up to $0.7dB$ and an average improvement of $0.36dB$. We outperform even the non-blind version of BM3D by an average of $0.25dB$ with our version trained on all noise levels and, when training only up to level $55$, we perform as well as the non-blind BM3D. Comparing the results of $\text{DnCNN}_{55}$ and of $\text{BUIFD}_{55}$ in Table 3.2, for unseen noise levels in the range $(55, 75]$, we see that the generalization of the fusion approach to unseen noise levels indeed applies to real images. The improvement of multiple PSNR points for level $75$ is consistent with that obtained in our synthetic experiment in Table 3.1.

The results in Table 3.3 illustrate denoising images with spatially varying noise levels, without re-training the networks. Noise is added across an image with a level that increases linearly with rows. For the non-blind BM3D, we input the average noise level as a guide. The BUIFD network can handle spatially varying noise that neither the prior nor the noise level predicting network branches are trained on. It outperforms DnCNN on all noise setups, whether the networks are trained on the full range or only up to level $55$.

For color image denoising, we use the standard color version of BSD68 (CBSD68) for testing. Noise is simulated and added to each test image before running it through a denoising method. PSNR results are reported in Table 3.4. The high inter-channel correlation between the RGB color channels [43] enables all methods to perform significantly better in terms of denoising PSNR on color images, compared with grayscale images. We note that this advantage of having multiple correlated channels, as in color imaging, is not always available: for instance, with single-wavelength imaging [84]. We hypothesize that this correlation also enables the networks to implicitly learn the noise-level prediction. High correlation means that the network sees multiple approximately equal data samples with different noise instances drawn from the same distribution. Thus, it more easily learns an estimate of the noise variance compared with the grayscale setup. Each of the two networks therefore performs more or less the same, when trained up to noise level $55$ and when trained up to noise levels $75$. Our fusion approach, however, consistently outperforms CDnCNN on every single noise level for both training noise ranges. Our average improvement over CDnCNN is about $0.1dB$. We also note that the networks outperform, on average, even the non-blind CBM3D by about $0.5dB$ for CDnCNN and $0.6dB$ for our CBUIFD.

Sample image denoising results for grayscale and color images are illustrated in Figure 3.4, 3.5 and Figure 3.6, 3.7, respectively, for the non-blind BM3D and the blind networks DnCNN and BUIFD trained on the full range of noise levels. The main trade-off seen between the results of BM3D and those of DnCNN is in the details of the reconstruction. The non-blind BM3D achieves good PSNR reconstruction but at the expense of blurring the results. This causes a loss of details (visible on the large rock in Figure 3.4, and the zoom-in insert in Figure 3.5) and a loss of edge sharpness (visible on the borders of the lake in the zoom-in insert in Figure 3.4). The DnCNN results suffer less of a blurring problem, but the noise removal is not optimal in certain areas such as smooth surfaces (visible on the inner area of the lake in the zoom-in insert

in Figure 3.4). Our approach achieves a good performance in terms of this trade-off. BUIFD achieves good PSNR results, with significantly less blurring than the non-blind BM3D (see Figure 3.5 for example).

### 3.4.5 Extended Benchmark Comparisons

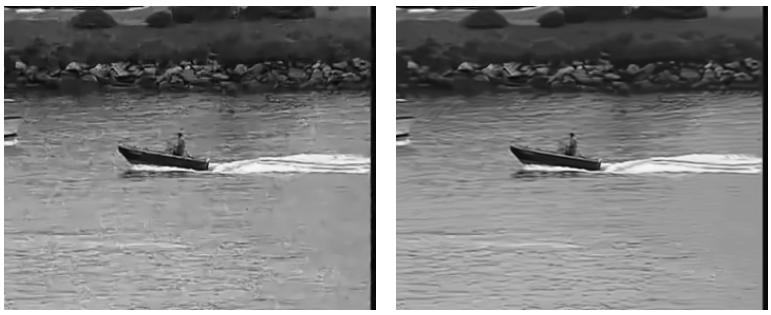
We present more denoising experimental tests on different benchmark datasets, and we compare the results of different denoising approaches on these datasets. We report *blind* denoising results for noise levels 10 to 80 (with a step size of 10) on the BSD68 dataset, Set14, Sun_Hays80, Urban100, and Manga109 datasets. Set14 comprises 14 images used traditionally for testing image-processing algorithms. Most of these images are smaller than $512 \times 512$. The Sun_Hays80 dataset is made up of the high-resolution version of the 80 images presented in [123], with sizes smaller than $1024 \times 1024$. The Urban100 dataset is a collection of 100 high-resolution images taken from Flickr by using urban keywords [60]. The Manga109 dataset is constituted of 109 professional artist drawings [92], of size $827 \times 1170$. We present in Tables 3.5 and 3.6 the denoising results of the blind non-learning methods BM3D, EPLL [165], KSVD [2], and WNNM [56] that were developed for Gaussian denoising. These methods are given, to enforce the blind setting, the default noise level set by the non-blind BM3D (set to 25). And we also present the results of the learning-based methods DnCNN [150] and BUIFD, on denoising the luminance of the images with added Gaussian noise levels ranging from 10 to 80. We also evaluate another learning-based method with the same training hyper-parameters as those of DnCNN, namely, the MemNet architecture [125]. We extend our fusion technique to this architecture and call it BUIFD(M). It is constructed following Figure 3.1(c), with the exception that the MemNet architecture replaces that of DnCNN for the prior-predicting CNN. All the learning-based methods in this section are trained up to noise level 55. Tables 3.5 and 3.6 show the PSNR and SSIM metrics for each method, and highlighted in bold are the best-PSNR and best-SSIM method between DnCNN and BUIFD, and between MemNet and BUIFD(M). A sample visual result is shown in Figure 3.8, taken from Set14.

## 3.5 Conclusion

In this chapter, we have defined a theoretical framework under which we derive an optimal denoising solution that we call fusion denoising. This theoretical setup enables us to study the statistical optimality of the network's learning, which is close to optimal for training noise levels but fails to generalize well for unseen noise levels. We integrate the fusion denoising approach into a deep-learning architecture to guide its learning and we compare with the optimal mathematical solution and with a state-of-the-art blind universal denoiser. Our synthetic experimental results show that our Fusion Net generalizes far better to higher unseen noise levels.

We have learned a data-dependent fusion function to adapt our fusion denoising network approach to real images. Our blind universal image fusion denoising network (BUIFD) improves the state-of-the-art real-image denoising performance, both on training noise levels and on unseen noise levels. This highlights the importance, both in terms of the interpretability and performance, of guiding the network towards learning more explicit priors and performing the proper integration with the data-fidelity terms. In this chapter, we have defined our prior, over the statistical distribution of pixel intensities rather than over the image space. In the following chapter, we extend this approach of prior decoupling to an image-space prior.

(a) Noisy image (level 25) (b) KSVD 27.59/0.691 (c) BM3D 28.3/0.758



(d) EPLL 28.18/0.743 (e) WNNM 28.21/0.748

(f) Ground-truth (g) DnCNN 28.13/0.77 (h) BUIFD **28.35/0.777**

(i) MemNet 28.03/0.77 (j) BUIFD(M) 28.29/0.774

Figure 3.8 – Sample visual result from Set14, with PSNR($dB$)/SSIM values. The top row shows *non-blind* results with the traditional methods KSVD, BM3D, EPLL and WNNM, as the noise level is 25, which is the default set when the noise level is unknown. And the bottom row shows the results with the different learning methods.

| Dataset | Method | 10 | 20 | 30 | 40 |
|---|---|---|---|---|---|
| BSD68 | KSVD [2] | 27.10/0.713 | 27.71/0.750 | 26.56/0.715 | 21.48/0.444 |
| | BM3D [29] | 29.18/0.802 | 28.69/0.798 | 27.35/0.763 | 22.44/0.495 |
| | EPLL [165] | 29.51/0.798 | 29.14/0.808 | 26.07/0.707 | 20.82/0.430 |
| | WNNM [56] | 27.83/0.750 | 28.35/0.779 | 27.06/0.743 | 21.89/0.468 |
| | DnCNN [150] | **33.61**/0.926 | 29.65/0.838 | 27.17/0.754 | 25.31/0.682 |
| | BUIFD | 33.58/**0.926** | **29.91/0.852** | **27.80/0.787** | **26.30/0.731** |
| | MemNet [125] | 33.33/0.927 | 29.59/0.848 | 27.32/0.769 | 25.63/0.701 |
| | BUIFD(M) | **33.59/0.928** | **29.90/0.856** | **27.83/0.794** | **26.36/0.740** |
| Set14 | KSVD [2] | 27.73/0.729 | 28.19/0.755 | 26.74/0.711 | 21.50/0.447 |
| | BM3D [29] | 30.58/0.832 | 29.68/0.818 | 27.85/0.772 | 22.55/0.502 |
| | EPLL [165] | 30.45/0.814 | 29.70/0.815 | 26.28/0.707 | 20.89/0.435 |
| | WNNM [56] | 28.89/0.777 | 29.24/0.796 | 27.49/0.748 | 22.00/0.476 |
| | DnCNN [150] | **33.81/0.914** | 29.98/0.832 | 27.39/0.757 | 25.40/0.688 |
| | BUIFD | 33.73/0.914 | **30.34/0.852** | **28.18/0.795** | **26.55/0.742** |
| | MemNet [125] | 33.45/0.912 | 29.91/0.842 | 27.43/0.767 | 25.56/0.701 |
| | BUIFD(M) | **33.70/0.914** | **30.29/0.854** | **28.19/0.801** | **26.62/0.751** |
| Sun_Hays80 | KSVD [2] | 28.80/0.767 | 29.22/0.778 | 27.21/0.681 | 21.45/0.374 |
| | BM3D [29] | 31.35/0.848 | 30.63/0.837 | 28.93/0.787 | 23.11/0.465 |
| | EPLL [165] | 31.09/0.826 | 30.76/0.833 | 27.16/0.710 | 21.12/0.380 |
| | WNNM [56] | 29.89/0.795 | 30.30/0.810 | 28.56/0.750 | 22.43/0.416 |
| | DnCNN [150] | 34.94/0.933 | 31.08/0.853 | 28.48/0.771 | 26.24/0.689 |
| | BUIFD | **34.99/0.935** | **31.44/0.871** | **29.36/0.814** | **27.77/0.763** |
| | MemNet [125] | 34.65/0.932 | 31.07/0.864 | 28.74/0.792 | 26.88/0.726 |
| | BUIFD(M) | **34.97/0.935** | **31.42/0.872** | **29.39/0.819** | **27.86/0.771** |
| Urban100 | KSVD [2] | 27.49/0.793 | 27.93/0.808 | 26.05/0.726 | 21.21/0.487 |
| | BM3D [29] | 30.98/0.884 | 29.93/0.868 | 27.87/0.818 | 22.64/0.565 |
| | EPLL [165] | 30.06/0.857 | 29.16/0.851 | 25.99/0.748 | 20.90/0.489 |
| | WNNM [56] | 28.03/0.796 | 28.54/0.805 | 27.16/0.768 | 21.93/0.529 |
| | DnCNN [150] | **34.10/0.935** | 30.01/0.870 | 27.10/0.797 | 24.76/0.723 |
| | BUIFD | 33.72/0.933 | **30.18/0.882** | **27.86/0.833** | **26.04/0.783** |
| | MemNet [125] | 33.46/0.930 | 29.65/0.869 | 26.89/0.799 | 24.81/0.734 |
| | BUIFD(M) | **33.63/0.933** | **30.03/0.881** | **27.73/0.832** | **25.99/0.785** |
| Manga109 | KSVD [2] | 29.91/0.871 | 29.69/0.868 | 27.08/0.763 | 22.02/0.519 |
| | BM3D [29] | 33.45/0.924 | 31.52/0.910 | 28.80/0.858 | 23.54/0.607 |
| | EPLL [165] | 33.31/0.915 | 31.29/0.905 | 27.22/0.795 | 21.73/0.531 |
| | WNNM [56] | 31.58/0.870 | 31.31/0.872 | 28.80/0.803 | 22.83/0.511 |
| | DnCNN [150] | 35.57/0.936 | 30.50/0.831 | 26.78/0.725 | 23.98/0.638 |
| | BUIFD | **35.88/0.947** | **31.86/0.907** | **29.09/0.864** | **26.92/0.820** |
| | MemNet [125] | 34.88/0.940 | 30.58/0.867 | 27.34/0.777 | 24.92/0.699 |
| | BUIFD(M) | **35.81/0.948** | **31.84/0.912** | **29.19/0.878** | **27.15/0.848** |

Table 3.5 – PSNR/SSIM evaluation of the *blind* BM3D, EPLL, KSVD, WNNM, DnCNN, BUIFD, MemNet, and BUIFD(M). Bold indicates the best denoising result in terms of PSNR or SSIM between each pair of learning methods, for different Gaussian noise levels, with clipped noisy images.

| Dataset | Method | 50 | 60 | 70 | 80 |
|---|---|---|---|---|---|
| BSD68 | KSVD [2] | 18.12/0.297 | 15.85/0.212 | 14.28/0.165 | 13.09/0.134 |
| | BM3D [29] | 18.19/0.316 | 15.73/0.220 | 14.10/0.166 | 12.90/0.132 |
| | EPLL [165] | 17.54/0.289 | 15.70/0.215 | 14.53/0.175 | 13.61/0.149 |
| | WNNM [56] | 18.22/0.306 | 15.83/0.217 | 14.21/0.166 | 13.00/0.134 |
| | DnCNN [150] | 23.75/0.616 | 22.29/0.546 | 19.42/0.352 | 16.67/0.233 |
| | BUIFD | **25.06/0.680** | **23.97/0.637** | **22.28/0.536** | **19.63/0.374** |
| | MemNet [125] | 24.35/0.646 | 23.34/0.606 | 21.53/0.499 | 18.43/0.320 |
| | BUIFD(M) | **25.15/0.690** | **24.14/0.655** | **22.14/0.537** | **18.92/0.363** |
| Set14 | KSVD [2] | 18.12/0.304 | 15.90/0.219 | 14.28/0.169 | 13.08/0.138 |
| | BM3D [29] | 18.24/0.326 | 15.82/0.231 | 14.14/0.172 | 12.91/0.137 |
| | EPLL [165] | 17.58/0.298 | 15.76/0.221 | 14.54/0.179 | 13.61/0.152 |
| | WNNM [56] | 18.28/0.316 | 15.92/0.227 | 14.24/0.173 | 13.01/0.138 |
| | DnCNN [150] | 23.66/0.625 | 22.09/0.553 | 19.35/0.364 | 16.54/0.239 |
| | BUIFD | **25.23/0.694** | **23.98/0.650** | **22.33/0.556** | **19.69/0.391** |
| | MemNet [125] | 24.17/0.649 | 23.06/0.609 | 21.34/0.506 | 18.29/0.331 |
| | BUIFD(M) | **25.33/0.706** | **24.19/0.670** | **22.27/0.557** | **19.08/0.388** |
| Sun_Hays80 | KSVD [2] | 18.05/0.233 | 15.82/0.161 | 14.24/0.122 | 13.03/0.097 |
| | BM3D [29] | 18.44/0.268 | 15.89/0.175 | 14.20/0.126 | 12.94/0.097 |
| | EPLL [165] | 17.68/0.236 | 15.79/0.167 | 14.57/0.133 | 13.63/0.110 |
| | WNNM [56] | 18.51/0.246 | 16.06/0.164 | 14.36/0.120 | 13.09/0.094 |
| | DnCNN [150] | 24.33/0.617 | 22.55/0.535 | 19.53/0.306 | 16.63/0.183 |
| | BUIFD | **26.41/0.716** | **25.14/0.674** | **23.24/0.561** | **20.17/0.364** |
| | MemNet [125] | 25.39/0.670 | 24.17/0.629 | 22.34/0.520 | 18.78/0.297 |
| | BUIFD(M) | **26.55/0.728** | **25.37/0.696** | **23.19/0.575** | **19.44/0.373** |
| Urban100 | KSVD [2] | 18.05/0.353 | 15.88/0.274 | 14.32/0.224 | 13.12/0.188 |
| | BM3D [29] | 18.43/0.387 | 15.97/0.287 | 14.30/0.227 | 13.04/0.189 |
| | EPLL [165] | 17.73/0.354 | 15.89/0.280 | 14.65/0.236 | 13.69/0.205 |
| | WNNM [56] | 18.33/0.367 | 15.98/0.268 | 14.33/0.208 | 13.08/0.168 |
| | DnCNN [150] | 22.87/0.656 | 21.17/0.579 | 18.84/0.414 | 16.41/0.303 |
| | BUIFD | **24.54/0.736** | **23.23/0.690** | **21.67/0.597** | **19.45/0.454** |
| | MemNet [125] | 23.25/0.679 | 22.10/0.638 | 20.61/0.547 | 18.17/0.396 |
| | BUIFD(M) | **24.57/0.742** | **23.36/0.704** | **21.64/0.602** | **19.00/0.455** |
| Manga109 | KSVD [2] | 18.70/0.355 | 16.42/0.245 | 14.73/0.187 | 13.45/0.152 |
| | BM3D [29] | 19.20/0.428 | 16.63/0.296 | 14.79/0.208 | 13.39/0.160 |
| | EPLL [165] | 18.34/0.365 | 16.31/0.252 | 14.93/0.198 | 13.87/0.167 |
| | WNNM [56] | 18.93/0.335 | 16.48/0.234 | 14.73/0.171 | 13.41/0.132 |
| | DnCNN [150] | 21.82/0.569 | 20.03/0.493 | 17.88/0.331 | 15.74/0.236 |
| | BUIFD | **25.13/0.777** | **23.58/0.731** | **21.89/0.632** | **19.63/0.480** |
| | MemNet [125] | 23.11/0.641 | 21.79/0.603 | 20.28/0.512 | 18.05/0.369 |
| | BUIFD(M) | **25.45/0.822** | **23.97/0.795** | **22.13/0.692** | **19.53/0.556** |

Table 3.6 – PSNR/SSIM evaluation of the *blind* BM3D, EPLL, KSVD, WNNM, DnCNN, BUIFD, MemNet, and BUIFD(M). Bold indicates the best denoising result in terms of PSNR or SSIM between each pair of learning methods, for different Gaussian noise levels, with clipped noisy images.

# Chapter 4

# Decoupling Learned Prior Hallucination and Data Fidelity in Image Restoration

In this chapter, we decouple the data fidelity and the learned prior components in image restoration, while exploiting the modeling strength of deep neural networks.

Classic image-restoration algorithms use a variety of priors, either implicitly or explicitly. Their priors are hand-designed and their corresponding weights are heuristically assigned. Hence, deep learning methods often produce superior image restoration quality. Deep networks are, however, capable of strong and hardly predictable hallucinations of the data to be restored. Networks jointly and implicitly learn to be faithful to the observed data while learning an image prior; and the separation of original data and hallucinated data downstream is then not possible. This limits their wide-spread adoption in image restoration applications. Furthermore, it is often the hallucinated part that is victim to degradation-model overfitting, as we show in Chapter 2.

We present an approach with decoupled network-prior based hallucination and data fidelity terms. We refer to our framework as the Bayesian Integration of a Generative Prior (BIGPrior). Our BIGPrior method is rooted in a Bayesian restoration framework and tightly connected to classic restoration methods. In fact, our approach can be viewed as a generalization of a large family of classic restoration algorithms. We use a recent network inversion method to extract image prior information from a generative network. We show that, on image colorization, inpainting and on denoising, our framework consistently improves the prior results through good integration of data fidelity. Our method, though partly reliant on the quality of the generative network inversion, is competitive with state-of-the-art supervised and task-specific restoration methods. It also provides an additional metric that sets forth the degree of prior reliance per pixel. Indeed, the per pixel contributions of the decoupled data fidelity and prior terms are readily available in our proposed framework.

---

Our code and models are made publicly available at https://github.com/majedelhelou/BIGPrior
This work is under review in the IEEE Transactions on Image Processing (TIP), 2021. [44]

## 4.1 Introduction

Image restoration recovers original images from degraded observations. It is based on two fundamental aspects, specifically, the relation to the observed data and the additional assumptions or image statistics that can be considered for the restoration. The relation to the observed data is referred to as *data fidelity*. The remaining information, brought in by the restoration method based on prior assumptions, is referred to as *prior hallucination*. It is termed hallucination because the added information is derived from a general model or assumption and might not faithfully match the sample image.

The data fidelity and prior terms emerge theoretically in the MAP formulation, but can also be implicitly induced by the restoration algorithms. For instance, non-local means [15] and BM3D [29] utilize the prior assumption that there exists different similar patches within an image. Diffusion [101] methods build on local smoothness assumptions. Data fidelity is typically enforced through the squared norm [71] that is equivalent to a MAP-based Gaussian noise model.

Classic image-restoration algorithms often rely on optimizations over explicit priors. An advantage of explicitly defined priors is the ability to easily control the relative relation between the weight of the data fidelity term and the weight ($\beta$) of the prior term. The general approach consists of an optimization

$$\arg\min_{x} \psi_d(f'(x), y) + \beta \cdot \psi_p(f''(x)), \tag{4.1}$$

where $y$ is the observation, $f'$ and $f''$ are various manipulation functions, $\psi_d$ enforces the data fidelity, and $\psi_p$ enforces the prior information. The optimal point is the estimate of the original image $x$. By making the prior term explicit, it is possible to have control over its contribution hence often better intuition and understanding of the reliability of the final restoration result. However, we note two shortcomings of these methods and we expand upon them in the following: (1) $\beta$ is not adapted based on the confidence in the fitness of the prior, and (2) the priors are hand-designed heuristics.

(1) The parameter $\beta$ should be inversely related to the quality of the observed degraded signal, but it should also be directly related to how well the assumed prior corresponds to the input image distribution or statistics. Although some methods, discussed in the section on related work, adjust their priors to the input data; they do not control $\beta$ based on the confidence in the fitness of the prior to the current sample. (2) Recent methods with implicit data-learned priors, notably relying on deep CNNs, outperform the classic methods with hand-designed priors on various image restoration tasks. This is due to the rich prior learned by discriminative networks or generative networks that, with adversarial training, can even learn image distributions to synthesize new realistic photos [65, 66, 67]. It is worth noting, however, that domain-specific prior information can still be explicitly enforced to improve the performance of the networks [45, 118].

One shortcoming of the deep learning methods is the loss of interpretability and control between data fidelity and prior-based hallucination. Given an image restored by a network, it is not possible to know how faithful it is to the observed signal versus how much prior-based hallucination was integrated in the image. And these hallucinations are not always reliable and can be prone to overfitting [46]. Hence, it is important to have a grasp of the prior hallucination taking place in the restoration process.

To obtain decoupled prior-based hallucination and data fidelity terms, we propose a novel framework

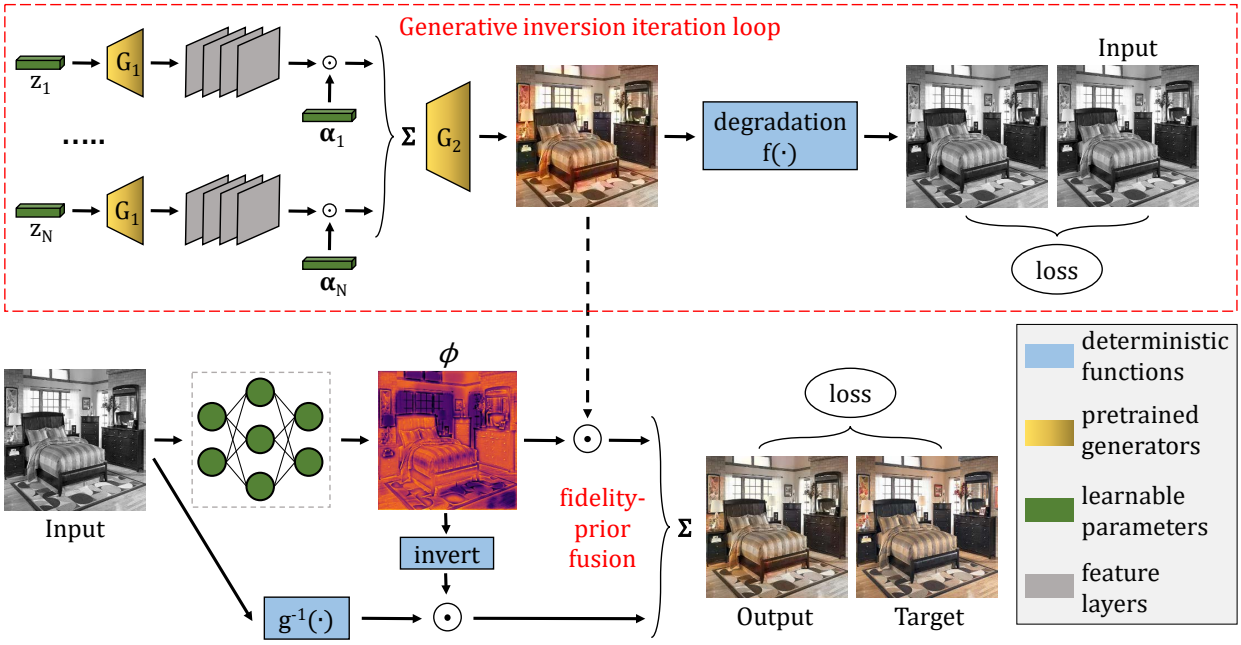



Figure 4.1 – Weights that are optimized are shown in green, and the sub-networks of the pre-trained generative network are shown in yellow. The generative network inversion process is optimized over a fixed set of iterations, which regularizes the output [131]. The final output is obtained through the fusion of the prior-based hallucination and the signal information, based on our $\phi$ map estimation.

that we call the **B**ayesian **I**ntegration of a **G**enerative **Prior** (BIGPrior framework). We replace the implicit data prior learned in feed-forward restoration networks with an explicit generative-network prior. This prior is then integrated following a MAP setting, where the data fidelity and prior terms are combined with a fusion weight that is adaptive to both. The BIGPrior framework is a generalization of a large family of classic restoration methods where the prior and its contribution weight are both learned, and the weight can adapt to both the signal quality *and* the fitness of the prior to the observed data.

Our framework structurally provides a reliable metric for per-pixel data fidelity in the final output to answer the question,"How much hallucination is there - at worst - in the output?". We present and analyze this metric by using blind denoising experiments. We also apply our method to various image restoration tasks and show consistent improvements, notably over the direct use of the generative prior, and we provide our faithfulness, a.k.a. data fidelity, metric.

## 4.2 Related Work

### 4.2.1 Classic Image Restoration

A variety of classic restoration methods, such as non-local means (NLM) [15], BM3D [29], their variants [30, 75] or combinations with sparse coding [35, 88], and diffusion-based methods [101, 25], make use of various prior assumptions on self-similarity or frequency-content distribution. Other algorithms formulate the prior

explicitly. For instance, dictionary-based methods [109] that assume images can be well represented by a fixed set of elements, which we discuss in the next section. Other examples are shrinkage methods [38, 138]. They can be directly connected with the family of MAP estimators, by deriving from the foundational work of Bayes and Laplace [73]. Considering an example with a hyper-Laplacian prior on image gradients, originally used in the context of deblurring in [71], optimizing the MAP negative log-likelihood

$$\arg\min_x -\log(P_{Y|X}(y|x)P_X(x)), \tag{4.2}$$

yields the estimator

$$\hat{x} = \arg\min_x ||x - y||_2^2 + \beta \cdot \sum_{j=1}^{J} |x \otimes f_j|^\gamma, \tag{4.3}$$

where $y$ is the signal we observe, $\hat{x}$ is the estimate of the target $x$, $\{f_j\}$ are $J$ first-order derivative filters and $\beta$ is a weight parameter. Setting $\gamma$ to one, with the corresponding filters, gives the special case of total-variation methods [110]. Generally these approaches are an optimization of the form

$$\hat{x} = \arg\min_x \psi_d(x, y) + \beta \cdot \psi_p(\mathcal{T}(x)), \tag{4.4}$$

where $\psi_d$ is the data-fidelity loss term, and $\psi_p$ incorporates the prior information on a transformation $\mathcal{T}$ of $x$ that could be the identity. $\mathcal{T}$ can also be based on derivatives [71], or wavelet [111] and other sparsifying transformations. For instance, WNNM [56] assumes that subsets of similar image patches are low-rank and uses a weighted nuclear norm for the low-rank minimization problem on similar patch groups. As with many classic image denoising methods, WNNM adapts $\beta$ based on the noise level and controls the data fidelity weight as such. However, as we noted in our introduction, these methods face two shortcomings. First, $\beta$ is not adapted based on the confidence in the prior given the degraded observation, but only on the quality of the latter. Hence, it is adapted based on the signal quality, such as the noise level, but also only following certain heuristics. Second, the prior itself is fixed based on hand-designed heuristics. We preserve an interpretable control over the contribution of the prior and decouple it from data fidelity, and we exploit learned network priors and increasing the flexibility in the fusion weight. Therefore, this weight is *learned* in our framework and can adapt both to the quality of the observed data, as well as the fitness of the prior, given the test observation.

### 4.2.2 Deep Neural Networks

The rich priors that convolutional neural networks are able to learn have improved image restoration results [63, 150, 155, 146]. These methods use sample-based learning and can extract prior information from large image datasets. This has enabled these deep learning methods to improve the state of the art on many restoration tasks [160]. However, the learned priors are implicit, meaning neither the prior nor its contribution can be disentangled from the data fidelity component in the final restored output. As recently shown for super-resolution and denoising tasks [46], networks can learn a frequency-conditional hallucination that is prone to overfitting to the training degradation models. Another recent example is in 3D reconstruction [126], where networks learn to recognize observations and to use memorized data samples, rather than to perform the reconstruction. In other words, the prior contribution can dominate over the data fidelity. Controlling

this trade-off is, however, not attainable within the neural networks. Our proposed framework enables us to exploit the strength of learned network priors and keep both control and insight over the data fidelity and prior trade-offs.

Extracting prior information from neural networks is possible through an inversion process [34, 4]. By searching the network-learned space of image distributions, it is possible to project on it in a fashion similar to dictionary-based methods. Generative networks are sufficiently powerful to be trained to learn different distributions, such as image or noise distributions [22]. A network inversion is carried out in [55], where it is used for different image processing tasks. We discuss this in more detail in the following section. A generative network inversion is also conducted in [96]. However, the method performs a fine-tuning of the pre-trained generator that goes against our objective to project on a fixed learned space. We also emphasize that our goal is not to improve such priors, rather to use them in our framework as image-projection spaces.

### 4.2.3 Signal Adaptation of Priors

As described in our discussion on classic methods, some of them [56] adapt the weight assigned to their prior term according to the quality of the observed signal. However, the fitness of the chosen prior can itself be image dependent. In other words, the prior can be accurate on certain images, but not as fit to be applied to others. Yet this is rarely accounted for in the literature. In the content-aware image prior presented in [26], although the weight of the prior itself is not adaptive, the hyper-Laplacian prior used is tweaked to adjust to the texture in the observed signal. Similarly, the method in [27] carefully selects its filters upon processing of the observed signal, hence altering its implicit priors. Also in the same spirit, some recent deep learning methods have tried to adapt to the observed inputs, through self-supervised weight modification [77], or novel learning [10]. This approach has even appeared in recent classification work to adjust to distribution shifts [124]. Such methods address the issue of the fitness of the prior to the given input by modifying the former on the fly. However, once a prior is selected, its fitness relative to the observed signal's quality is dismissed. The weight of the prior term is therefore not adaptive, and the prior's contribution cannot be decoupled from data fidelity.

## 4.3 Method

In designing our method, we address the shortcomings discussed in the introduction. We present a framework where the prior and the data-fidelity terms are explicit. This enables us to exploit the modeling strength of deep neural networks for the prior and enables us to learn a weight between the prior and the data fidelity that is doubly adaptive to the quality of the observation and to the fitness of the prior to the input's distribution. Rather than combining the contribution of the prior and the data-fidelity terms through an optimization, we explicitly enforce their fusion in the final output. This explicit decoupling of the two terms enables us, as well as downstream applications, to gain in restoration interpretability. In this section, we present the mathematical details of our proposed method and its relation to classic families of restoration algorithms. We also present a network-based prior that relies on generative-network projection and introduce our approach for learning the adaptive weight without guided supervision.

### 4.3.1 Mathematical Formulation

Given an observed signal $y$ that is a degraded version of the image $x$, our restoration estimate $\hat{x}$ is formulated as

$$\hat{x} = \underbrace{(1 - \phi(y; \theta_1)) \odot g^{-1}(y)}_{data\ fidelity} + \underbrace{\phi(y; \theta_1) \odot G(z^*; \theta_2)}_{prior}, \tag{4.5}$$

where $g^{-1}(\cdot)$ is a bijective function that we discuss in what follows, $\phi(\cdot; \theta_1)$ is an estimator for the fusion factor, parameterized by $\theta_1$, and that assigns adaptive weights to the prior-based hallucination and the data fidelity. It is a generalization of $\beta$ that we learn internally from sample-based training. $G(z^*; \theta_2)$ is the prior-based hallucination, parameterized by $\theta_2$, described in detail in the following, and $\odot$ is the pixel-wise multiplication operator. To ensure a very strict lossless data-fidelity term, we restrict $g^{-1}(\cdot)$ to the set of bijective functions. We can choose it such that $g(\cdot)$ is close to the degradation model $f(\cdot)$ of the restoration task, as described in our experimental setup. We note that this formulation is closely related to the classic restoration methods based on explicit prior optimizations discussed in our related work. The difference is that our prior is based on a trainable neural network $G$, and that our fusion factor is also learned to be adaptive, per sample, both to the quality of the observed data and to the fitness of the prior.

We present the **relation to MAP estimation** in connection with the work in [45]. The authors derive a MAP estimate for additive white Gaussian noise removal where the additive noise ($y_i = x_i + n_i$) follows the normal distribution $\mathcal{N}(0, \sigma_n)$, and an explicit image prior is enforced. More precisely, the solution is derived with the assumption of a Gaussian prior [106] *on the pixel distribution*. With this model, the prior distribution for a pixel value $x_i$ follows $\mathcal{N}(\bar{x}_i, \sigma_{x_i})$, and this yields a MAP estimate

$$\hat{x}_i = \arg\max_{x_i} P_{X_i|Y_i}(x_i|y_i) = \frac{y_i}{1 + 1/S_i} + \frac{\bar{x}_i}{1 + S_i}, \tag{4.6}$$

with $S_i$ being the signal-to-noise ratio defined as

$$S_i \triangleq \frac{\sigma_{x_i}^2}{\sigma_n^2}. \tag{4.7}$$

Note how $S_i$ is, in fact, dependent on signal quality (through $\sigma_n$), as well as the confidence in the prior (through $\sigma_{x_i}$). Indeed, intuitively the larger $\sigma_{x_i}$ is, the less reliable the prior term $\bar{x}_i$ is; and the smaller it is, the more reliable the prior term is. In this special case of our general formulation,

$$\phi(y_i) = \frac{1}{1 + S_i}, \tag{4.8}$$

$g(\cdot)$ is the identity mapping, and the prior is the expected value over the distribution of the input $\mathbb{E}_{X_i}[x_i]$. Our formulation in Equation (4.5) generalizes this solution to non-Gaussian, as well as image-wise prior distributions, while taking into account signal quality and prior confidence.

We also describe the **relation to dictionary-based** methods. Dictionary-based methods [52, 109] generally follow the formulation

$$\hat{x} = \arg\min_{x, d(x, Dv) < \epsilon} \psi_d(x, y) + \beta \cdot \psi_p(v), \tag{4.9}$$

| | $\phi$ explicitly known | $\phi$ relation to *data fidelity* |
|---|---|---|
| Colorization | ✗ | Luminance and edge related |
| Inpainting | ✓ | Binary mask based |
| Denoising | ✗ | Noise-level adaptive |

Table 4.1 – The $\phi$ map values are only explicitly known for inpainting, but are always related to the data-fidelity and prior-confidence terms discussed in our mathematical formulation. Indeed, in colorization there exist strong relations between luminance and the fidelity of the observed data, in inpainting this directly matches the applied mask, and in denoising the noise level determines the fidelity of the observation. The $\phi$ map also, across all tasks, depends on the confidence in the prior.

| Method | Bedroom set AuC [155] ↑ | Church set AuC [155] ↑ |
|---|---|---|
| Colorful colorization [155] | <u>88.55</u> | 89.13 |
| Deep image prior [131] | 84.33 | 83.31 |
| Feature map opt. [10] | 85.41 | 86.10 |
| mGAN prior [55] | 88.52 | <u>89.69</u> |
| Ours | **89.27** | **90.64** |

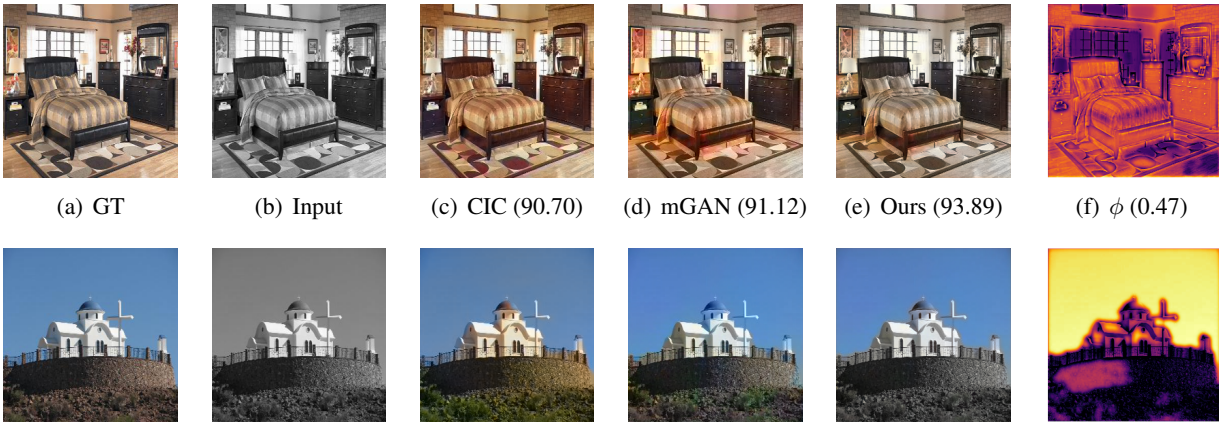
Table 4.2 – Quantitative AuC (%) results for image colorization on the Bedroom and Church test sets. The higher the value is, the lower the cumulative colorization error curve is. We highlight, with background shaded in gray, the widely used task-specific supervised method. The best results are shown in bold, and the second best are underlined.

where $D$ is the dictionary, specifically, a vector set that spans the dictionary space, $v$ holds the coordinates of a point in that space, $d(\cdot, \cdot)$ is a distance function, and $\epsilon$ is a small value in $\mathbb{R}_+$. It is typical to use a $\psi_p$ that encourages sparsity, thus to assume that the dictionary captures the main directions of variation in an image. This sparsity of $v$ parallels restrictions on the generative latent space. Effectively, enforcing

$$d(x, Dv) < \epsilon \tag{4.10}$$

is a subtle relaxation of the constraint $x \in \text{span}(D)$, which enforces the prior assumption that the image must belong to the dictionary space. This would correspond in our formulation of Equation (4.5) to $x \in \text{span}(G)$, where in our case the dictionary space is instead the learned space of a generative network. In our formulation, the restriction is enforced only on our decoupled prior element, rather than having to enforce it on $x$ itself and then relaxing it through a tweaking of $\epsilon$.

In summary, our formulation can be viewed as a general framework of MAP estimation and as a generalization of dictionary-based methods. We choose a strict data-fidelity term that preserves a bijective relation to the observed signal, and a fusion factor that takes into account both signal quality and prior confidence. The following two sections discuss in more detail the prior term and the $\phi$ fusion factor learning.



(a) GT (b) Input (c) CIC (90.70) (d) mGAN (91.12) (e) Ours (93.89) (f) $\phi$ (0.47)

(g) GT (h) Input (i) CIC (94.17) (j) mGAN (93.87) (k) Ours (94.75) (l) $\phi$ (0.52)

Figure 4.2 – From left to right are the ground-truth image (GT), the grayscale input, the results of colorful image colorization (CIC) [155], mGAN [55], and ours, with the AuC (%), and our channel-averaged $\phi$ map (with global average between parentheses). The darker colors indicate values of $\phi$ closer to 0, whereas bright yellow indicates those closer to 1.

### 4.3.2 Generative-Space Projection Prior

Theoretically, an inference method can be used to replace the prior term. For instance, a feed-forward network's output can replace $G(z^*)$ in Equation (4.5). However, such a network trained with supervision takes into account both the data-fidelity and prior terms, albeit without any insight as to how much prior-based hallucination occurs or any control over the different contributions. Therefore, in order to best decouple data fidelity from prior hallucination, we opt for a pre-trained generative network inversion to act as the learned prior. Effectively, this is a better strategy for decreasing the upper bound on a worst-case hallucination contribution. The inversion produces a sampling from the generative space, or a projection on that space as in dictionary-based projections discussed earlier. The latent code $z^*$ for the generative-space projection is obtained as

$$z^* = \arg\min_z \mathcal{L}_G(f(G(z)), y), \tag{4.11}$$

where $\mathcal{L}_G$ can be a weighted average of $\ell_1$, $\ell_2$, and perceptual losses, and $f(\cdot)$ is the degradation model of a restoration task. When using a single latent code, very limited information can be encoded, which yields a coarse prior, notably for high-resolution images. To avoid this loss of expressiveness, we use the recent multi-code GAN inversion method that splits the generative network $G$ into two stages, at layer $l$ [55]. The first stage $G_1^{(l)}$ generates multiple feature space representations, each corresponding to one of $N$ latent codes $\{z_n^*\}_{n=1}^N$, where every $\alpha$ is a vector of length equal to the number of feature-space channels. The second stage $G_2^{(l)}$ generates the output image based on a fused feature representation by using adaptive channel weights $\{\alpha_n^*\}_{n=1}^N$. The latent codes and adaptive weights are obtained, as in Equation (4.11), by an inversion optimization

$$\{z_n^*\}_{n=1}^N, \{\alpha_n^*\}_{n=1}^N = \arg\min_{\{z_n\}_{n=1}^N, \{\alpha_n\}_{n=1}^N} \mathcal{L}(f(x^{inv}), x), \tag{4.12}$$

| Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|---|---|---|---|
| DeepFill v2 [145, 146] | **26.56** | **0.9555** | **0.0191** |
| Feature map opt. [10] | 14.75 | 0.4563 | - |
| Deep image prior [131] | 17.92 | 0.4327 | - |
| mGAN prior [55] | 20.55 | 0.5823 | 0.2070 |
| Ours | <u>25.32</u> | <u>0.9240</u> | <u>0.0376</u> |

Table 4.3 – Quantitative PSNR ($dB$), SSIM, and LPIPS results for central image inpainting. We mask out a $64 \times 64$ patch from the center of each input image. The task-specific state-of-the-art method is highlighted with background shaded in gray. The best results are shown in bold, and the second best are underlined.

where the inverted image $x^{inv}$ is given by

$$G(z; \alpha, \theta_2) \triangleq x^{inv} = G_2^{(l)} \left( \sum_{n=1}^{N} G_1^{(l)}(z_n) \cdot \alpha_n \right). \tag{4.13}$$

Our image prior term in Equation (4.11) is then given by $G(z^*; \alpha^*, \theta_2)$, where $\theta_2$ are the frozen weights of the generative sub-networks $G_1$ and $G_2$. We also note that randomly traversing the latent space of a generative network can potentially produce hallucinated images that lie outside the natural image manifold [93]. This is averted by the guided inversion loss that maps the generative output, through the degradation model, to the observed image. The case of the generative projection being outside the natural-image manifold, which can occur when the degradation is extreme, still does not pose an issue in our framework. Indeed, this projection is already treated in our approach as a prior hallucination that might not be faithful to the original image.

### 4.3.3 Guide-Free $\phi$ Learning

A guided learning of the parameters $\theta_1$ to predict $\phi$ is possible for a task such as inpainting but impossible for other tasks. This is simply because inpainting is the extreme case where signal quality is binary, specifically zero at the masked areas. For other tasks, a target $\phi$ cannot be readily obtained. We thus train a network with weights $\theta_1$ to predict $\phi$ in an end-to-end manner, with $\phi$ effectively being an intermediate feature space having no explicit learning loss. Our mini-batch training loss $\mathcal{L}(x, y; \theta_1)$ for learning $\theta_1$ is given by (we use $\phi$ to also denote the network outputting it, for better readability)

$$\begin{aligned} \mathcal{L}(x, y; \theta_1, \theta_2) = || \, (1 - \phi(y, \theta_1)) \odot g^{-1}(y) \\ + \phi(y, \theta_1) \odot G(z^*; \alpha^*, \theta_2) - x||_2^2 + \rho \cdot ||\phi(y, \theta_1)||_1, \end{aligned} \tag{4.14}$$

where $\rho$ is a scalar weight that we discuss next, and the parameters $\theta_2$ of the generative network are the frozen weights of a *pre-trained* generative network. This end-to-end training enables the network predicting $\phi$ to learn to assess, based on the observation $y$, the quality of that observed image, as well as the fitness of the prior to this observation.

**Fidelity-Bias Balance.** For certain image test cases, the quality of the data-fidelity term can be very

| Test | Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|---|---|---|---|---|
| Bed. | mGAN prior [55] | 20.34 | 0.5902 | 0.2134 |
| | Ours | **23.22** | **0.8598** | **0.0775** |
| Chu. | mGAN prior [55] | 19.33 | 0.5359 | 0.2235 |
| | Ours | **21.94** | **0.8509** | **0.0855** |
| Conf. | mGAN prior [55] | 19.38 | 0.5641 | 0.2062 |
| | Ours | **22.20** | **0.8318** | **0.0785** |

Table 4.4 – Quantitative PSNR ($dB$), SSIM, and LPIPS results for randomized-masking inpainting on the Bedroom, Church (Outdoor), and Conference test sets. The randomized masking increases the difficulty of predicting our $\phi$ maps. To analyze the effect of mask randomization on the performance of our $\phi$ prediction compared to the central inpainting task, we compare the prior-based results to ours.

similar to that of the learned prior, at least over some subsets of pixels. This would induce no change in the loss term for varying values of our fusion factor $\phi$, as all would result in similar final outputs. However, for these cases, it is not necessary to *hallucinate* information as the data fidelity is also just as accurate. Therefore, we address these edge cases by adding an auxiliary loss on the $\ell_1$ norm of $\phi$ in Equation (4.14), which can additionally regularize the feature learning process [41]. This term enforces that the training favors smaller values of $\phi$ such that the overall contribution of the data fidelity term is maximized when this is not detrimental to the quality of the final output. This fidelity-bias term is weighted by the scalar $\rho$ in Equation (4.14).

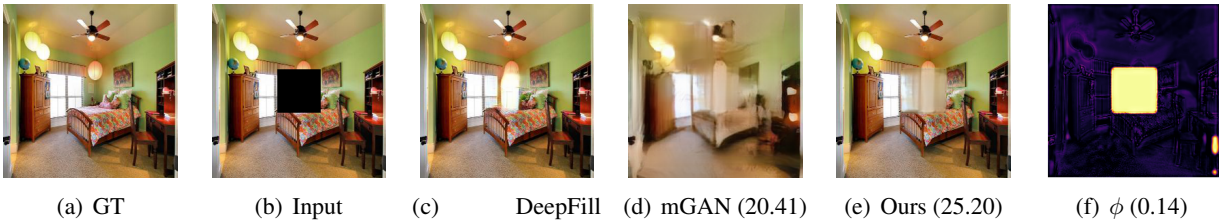
## 4.4 Experiments

We conduct experiments on image colorization, inpainting, and blind AWGN removal. Colorization does not induce an explicit solution for $\phi$, aside for certain exceptions that we discuss in the next section, such as edges and extreme luminance areas. Inpainting induces an explicitly known solution for $\phi$. Whereas, AWGN does not have an explicit solution for $\phi$, as the image prior is not explicitly formulated. However, the AWGN experimental setup enables us to analyze the guide-free learning of $\phi$, which would intuitively fluctuate mainly with the noise level (direct relation), but also marginally with the uncertainty in the prior (opposite relation), as described in Section 4.5. This is summarized in Table 4.1 and discussed in the following sections.

### 4.4.1 Experimental Setup

As described in Section 4.3, we use the multi-code GAN inversion approach for our generative-space projection prior. The pre-trained GAN models, which correspond to each dataset used, are all different versions of the PGGAN [65] network. They are pre-trained on the Bedroom, Church (Outdoor), and Conference room datasets taken from the LSUN database [144]. The details for each experiment follow the

mGAN [55]: https://github.com/genforce/mganprior
PGGAN [65]: https://github.com/tkarras/progressive_growing_of_gans



(a) GT (b) Input (c) DeepFill (21.88) (d) mGAN (20.41) (e) Ours (25.20) (f) $\phi$ (0.14)

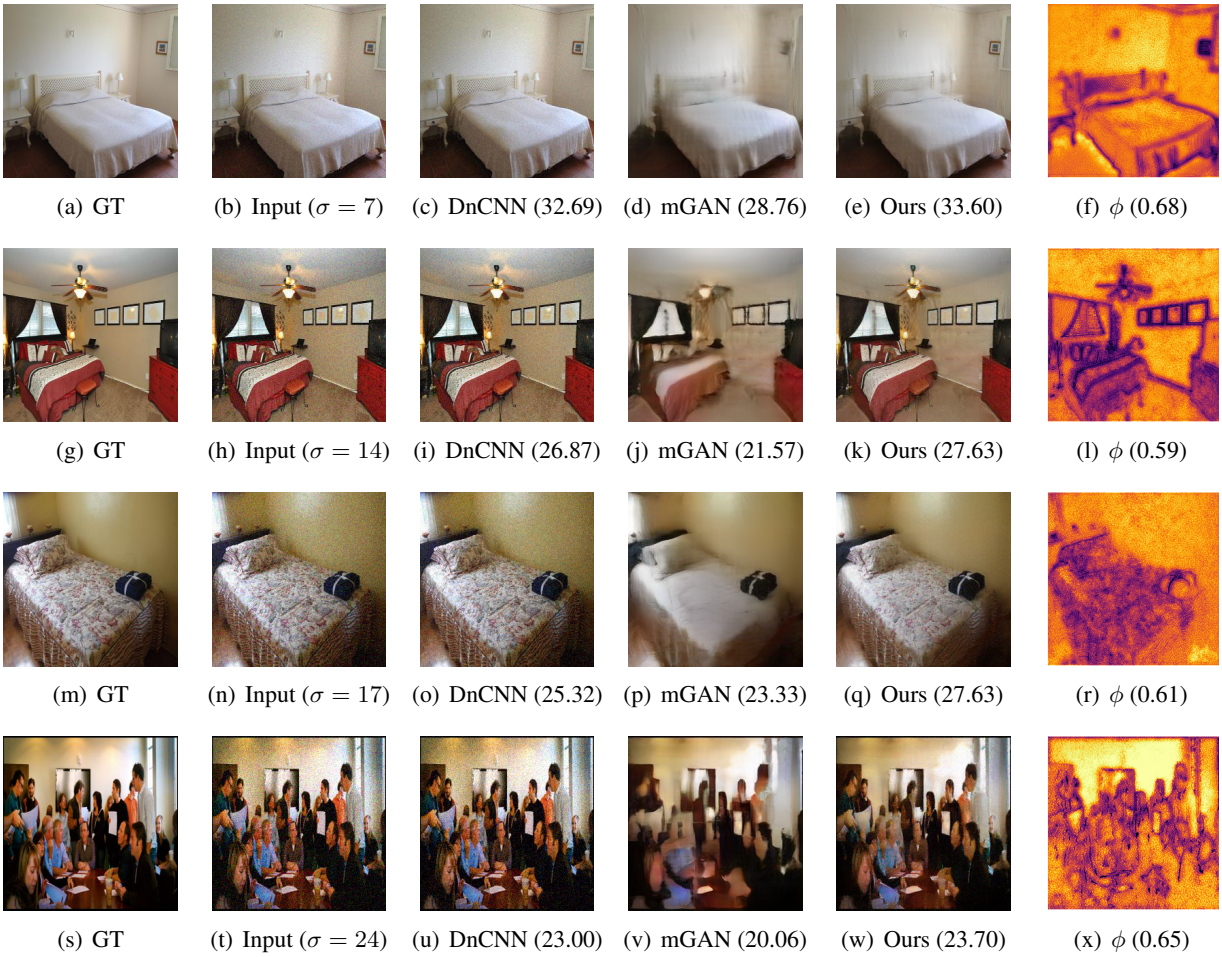
(g) GT (h) Input (i) DeepFill (29.38) (j) mGAN (21.45) (k) Ours (24.59) (l) $\phi$ (0.13)

(m) GT (n) Input (o) DeepFill (17.95) (p) mGAN (16.04) (q) Ours (17.98) (r) $\phi$ (0.22)

Figure 4.3 – From left to right are the ground-truth image (GT), the masked input, the results of DeepFill v2 [145, 146], mGAN [55], and ours, with the PSNR in $dB$, and our channel-averaged $\phi$ map (with global average between parentheses). The first two rows show example images from the standard central-inpainting benchmark, and the third row is an example from our randomized-inpainting experiment.

settings given by the authors of [55] and are given in the following sections. We note that any generative network, such as DCGAN [104], LR-GAN [141], CVAE-GAN [8], StyleGAN [66], StyleGAN2 [67], or even any future method allowing projections or sampling from a learned image distribution, can be used for the projection prior of our method. To enable direct comparisons with with mGAN [55], we use the PGGAN in our experiments. We use AuC [33, 155], PSNR, SSIM [135], and the perceptual metric LPIPS [156] in our quantitative evaluations.

For our fusion factor learning, we train the same backbone network with the same settings for all of our experiments. The architecture is inspired by [150] and is a residual learning made up of a sequence of convolutional, batch normalization, and ReLU blocks. We omit further architecture details that can be found in our code. We use a batch size of 8, a starting learning rate of 0.01, and a fidelity-bias balancing weight $\rho = 1e - 5$. We train for 25 epochs with random shuffling and update the learning rate following a cosine annealing with warm restarts scheduler [85]. The restart period is adaptive to the batch size such that it is always 4 epochs. We also note for reproducibility that training with images that are normalized to $[0, 1]$ and then zero-centered is empirically observed to improve the final results. The same normalization is then performed before inference and inverted once the output is obtained. We train our model with the loss of Equation (4.14) on a subset of the LSUN validation set that corresponds to each of the large training sets used for pre-training the PGGAN models, and we test on the remaining subset.



(a) GT (b) Input ($\sigma = 7$) (c) DnCNN (32.69) (d) mGAN (28.76) (e) Ours (33.60) (f) $\phi$ (0.68)

(g) GT (h) Input ($\sigma = 14$) (i) DnCNN (26.87) (j) mGAN (21.57) (k) Ours (27.63) (l) $\phi$ (0.59)

(m) GT (n) Input ($\sigma = 17$) (o) DnCNN (25.32) (p) mGAN (23.33) (q) Ours (27.63) (r) $\phi$ (0.61)

(s) GT (t) Input ($\sigma = 24$) (u) DnCNN (23.00) (v) mGAN (20.06) (w) Ours (23.70) (x) $\phi$ (0.65)

Figure 4.4 – From left to right are the ground-truth image (GT), the noisy input with the AWGN standard deviation, the results of DnCNN [150], mGAN [55], and ours, with the PSNR in $dB$, and our channel-averaged $\phi$ map (with global average between parentheses).

### 4.4.2 Colorization

For the colorization of a grayscale input image, unlike inpainting for example, it is much less predictable what an ideal $\phi$ map would be. We conduct colorization experiments, where the grayscale input is the luminance channel, and we evaluate the error on the $ab$ color space. The AuC metric [33, 155] computes the area under the cumulative percentage $\ell_2$ error distribution curve in the $ab$ space. The percentage is that of pixels lying within an error threshold that is swept over $[0, 150]$ in steps of one. For generative network inversion, we use the sixth layer of the PGGAN for the feature composition, with 20 latent codes, and $\ell_2$ and VGG-16 perceptual loss [117], optimized with gradient descent for 1500 iterations, following [55]. Our $g^{-1}(y)$ function duplicates the grayscale channel over each of the color channels. The remaining details follow the experimental setup of Section 4.4.1

We present our quantitative image colorization results in Table 4.2, along with those of the deep image prior [131], the feature map optimization [10], the colorful image colorization [155], which is a feed-

| Test | Method | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|---|---|---|---|---|
| Bed. | DnCNN† [150] | 24.96 | 0.5804 | 0.1859 |
| | mGAN prior [55] | 22.72 | 0.6257 | 0.1978 |
| | Ours | **26.80** | **0.7279** | **0.0998** |
| Church | DnCNN† [150] | 22.40 | 0.5166 | 0.2046 |
| | mGAN prior [55] | 21.12 | 0.5643 | 0.2065 |
| | Ours | **23.38** | **0.5959** | **0.1435** |
| Conf. | DnCNN† [150] | 22.81 | 0.5310 | 0.2167 |
| | mGAN prior [55] | 21.49 | 0.5962 | 0.1968 |
| | Ours | **24.70** | **0.6578** | **0.1192** |

Table 4.5 – PSNR ($dB$), SSIM, and LPIPS results for AWGN removal on the Bedroom, Church, and Conference sets. The noise follows a Gaussian distribution with standard deviation sampled uniformly at random from [5,50] per image. †We retrain and test DnCNN with the same data and setup as ours.

forward method supervised specifically for colorization, and the mGAN prior [55]. We note the considerable improvement of our method, despite the restriction of enforcing a strict data fidelity.

Visual results are shown in Figure 4.2 for the different image colorization methods. We can observe that $\phi$ is lower on image edges, which indeed generally constitute information that is not lost by the grayscale degradation. We observe as well that $\phi$ tends to be low when the luminance is around extreme values, as in such cases the grayscale images are faithful to the original color images. In both of these cases, it is the confidence in the data fidelity that is adapted to the observation. We also note, for instance in the sample of the second row, that $\phi$ can be very insightful. It indicates that the color of the sky was heavily hallucinated, whereas the bottom half and the church dome use almost no prior hallucination. This is advantageous for downstream tasks as the dome was, in fact, incorrectly hallucinated by the generative-network projection prior. This is similar for the grass, where green was incorrectly added.

Visual results are shown in Figure 4.3 for the different methods. For our method, there is little flexibility in terms of the fusion factor $\phi$ for the inpainting tasks, which are tasks with binary degradation, i.e., the signal is either perfectly available or not at all. The $\phi$ map effectively predicts the inpainting mask, a mask that is taken as input in the DeepFill method, and the quality of our results is tied mostly to those of the generative-network inversion, as can be visually observed.

### 4.4.3 Inpainting

We present results on the standard central-crop inpainting task in Table 4.3. A $64 \times 64$ patch is masked from the test image, and the task is to recover the hidden crop. For generative-network inversion, we use the fourth layer of the PGGAN for the feature composition, with 30 latent codes, and $\ell_2$ and VGG-16 perceptual loss [117], optimized with gradient descent for 3000 iterations, following [55]. We use an identity function $g^{-1}(y) = y$ for the data fidelity, and the remaining setup follows that of Section 4.4.1. The PSNR, SSIM and LPIPS results show a significant improvement of our approach, due to the use of the data fidelity, over the mGAN prior results ($+4.77dB$ in terms of PSNR). The inpainting results are averaged across

the Bedroom, Church (Outdoor), and Conference datasets. We compare them with the deep-image prior method [131] and with the recent feature map optimization approach [10] that is a method using GAN priors with test-image specific adaptation. For reference, we compare the results with a task-specific supervised inpainting method, namely, the most recent version [146] of DeepFill [145], trained on the Places2 dataset. DeepFill takes the mask as input and uses gated convolutions to account for invalid pixel locations, and contextual attention [145] to exploit similar patches across the image. The output is refined by using an adversarial GAN loss on every neuron in the feature space [146]. For inpainting, our approach cannot use anything out of the signal over the masked area hence is dependent on the prior hallucination.

The aforementioned benchmarking setup, however, makes the task simpler for our method in terms of predicting $\phi$. Therefore, we design a randomized-masking inpainting setup and present experimental results on it in Table 4.4. Our randomized-masking algorithm selects uniformly at random a number of patches to be masked, in $[2, 4]$. Then, per patch, a random pixel location for the corner of that patch is selected. The algorithm samples from a normal distribution $\mathcal{N}(64, 32)$, truncated to $[9, +\infty)$, a width and a height for each patch, with re-sampling in case the patch extends beyond the image coordinates. We compare the mGAN prior results with ours in Table 4.4. We omit the other methods because the purpose of this randomized-masking experiment is specifically to analyze the effect of randomizing the mask on our $\phi$ prediction, and to analyze how the incurred errors in $\phi$ affect the performance relative to the prior. We can first note that the mGAN performance decreases, by almost $0.02$ SSIM points on average. With the randomization of the mask, our performance decreases more significantly, by almost $0.1$ SSIM points, but still significantly improves over the mGAN results. This comparison highlights the increased difficulty of our internal $\phi$ prediction when the mask is randomized relative to the central inpainting task where the mask location is immutable.

### 4.4.4 Blind Denoising

We conduct experiments on blind denoising, specifically on AWGN removal. For blind denoising, we follow the standard setup [150, 45, 46] of sampling a noise level, uniformly at random over the range $[5, 50]$. This level is the standard deviation of the AWGN. For generative network inversion, we use the fourth layer of the PGGAN for the feature composition, with 30 latent codes, and $\ell_2$ and VGG-16 perceptual loss [117], optimized with gradient descent for 3000 iterations, following [55]. We set $f(\cdot)$ (Equation (4.11)) to the identity. Our $g^{-1}(y)$ function is also the identity function as the noise is zero-mean. Generally, $g^{-1}(\cdot)$ can be the subtraction of the noise mean value. For the remaining setup details, we follow the experimental setup of Section 4.4.1.

We present the AWGN removal results in Table 4.5, along with those of DnCNN [150], which we retrain on the same data as ours. Our approach achieves the best performance, consistently across the different datasets and evaluation metrics.

Visual results are shown in Figure 4.4 for the different methods. We observe that DnCNN preserves details well, but at the cost of poorer denoising on low-frequency regions (e.g., walls). The mGAN results are worse than DnCNN, but with our framework the final results become more visually pleasing and accurate. The $\phi$ map illustrates, per pixel, the contribution of hallucination relative to data fidelity and is again lower around edges, as with colorization. We analyze $\phi$ in more detail, in the context of AWGN removal, in the

next section.

## 4.5 Discussion

The AWGN experiments provide the ideal setup for an analysis of $\phi$ that we carry out in this section. We know that $\phi$ should be inversely related to the signal quality, the poorer the signal is, the higher the $\phi$ values are. And $\phi$ is then also directly related to the confidence in the prior, or the fitness thereof. With AWGN, the quality of the signal is also inversely related to the noise level, in this case, to the standard deviation of the Gaussian noise. We analyze the correlation between the mean $\phi$ value for a test image, and the standard deviation of the noise in this test image. Results are shown in Figure 4.7(a), with the Pearson correlation factor, for three datasets. We can clearly observe the positive correlation between the two variables, with a factor of 0.83 and 0.81 for the bedroom and conference sets, respectively. The correlation is lower, at 0.6, for the church dataset. The remaining factor of variability in $\phi$ is the fitness of the prior, which we analyze in Figure 4.7(b). The correlation between the average $\phi$ value and the generative PSNR is the highest for the church set, reaching 0.56, and supporting our claims with regard to $\phi$. Indeed, we observe that $\phi$ is well-correlated with the signal's quality, and when that correlation is somewhat lower it is matched with a higher correlation between $\phi$ and the fitness of the prior, exactly as expected from the MAP framework's perspective. To summarize, we make two supporting observations from our aforementioned analysis. First, the $\phi$ estimation, which is learned with no guide in our framework, strongly correlates with the signal quality. Second, a lower correlation with signal quality, as in the church set, is directly justified by a higher correlation between $\phi$ and the fitness of the prior to the test data. These two observations align exactly with the intuitions derived from the MAP estimation framework, as presented in Section 4.3.1.

The framework we present can be a novel basis for image restoration as it can counter the obstacle of degradation-model overfitting, common in image-restoration tasks. This is because hallucination is the key part prone to overfitting to the chosen model. Our framework can guard against this type of overfitting by relying on decoupled data fidelity and prior hallucination, and by using a pre-trained and frozen generative network, independent of the degradation model, for the hallucination part. Our fusion factor could also be used to increase the robustness and reliability of down-stream computer vision tasks, by making the latter aware of the extent of per-pixel hallucination in the restoration result. For instance, when a computer-vision algorithm deals with degraded images, rather than training the downstream network only on the restoration output, further information regarding the degree of hallucination can be used to increase robustness, notably against adversarial attacks. Our fusion map $\phi$ conveys such hallucination information, which can also be used for better interpretability of the results by human users.

## 4.6 Conclusion

We have presented a framework for image restoration that enables the use of deep networks for extracting an image prior while decoupling prior-based hallucination and data fidelity. We have shown how our framework is a generalization of a large family of classic restoration methods, notably of Bayesian MAP estimation setups such as the one presented in Chapter 3, and of dictionary-based restoration methods. We

have conducted experiments on image colorization, inpainting, and Gaussian denoising. Our results, which structurally come with a pixel-wise map indicating data fidelity versus prior hallucination contributions, outperform prior-based methods and are even competitive with state-of-the-art task-specific supervised methods. We have also presented an analysis of this fusion factor $\phi$ estimation that supports our different claims.

As we show in the failure cases, one of the current limitations of our framework is the quality of the generative-network prior modeling (Figure 4.5) and that of the network inversion for projecting on the learned-prior space (Figure 4.6). Another drawback, which we discuss in the following section on future work, is that the fusion between the data-fidelity and the prior-based terms is limited to the spatial domain. The fusion of the two terms is carried out pixel-wise, but could theoretically be conducted over the frequency domain or a combination of the two. This could be of interest for restoration tasks where the degradation model is not pixel based and where the learning, as we note in Chapter 2, is frequency based.

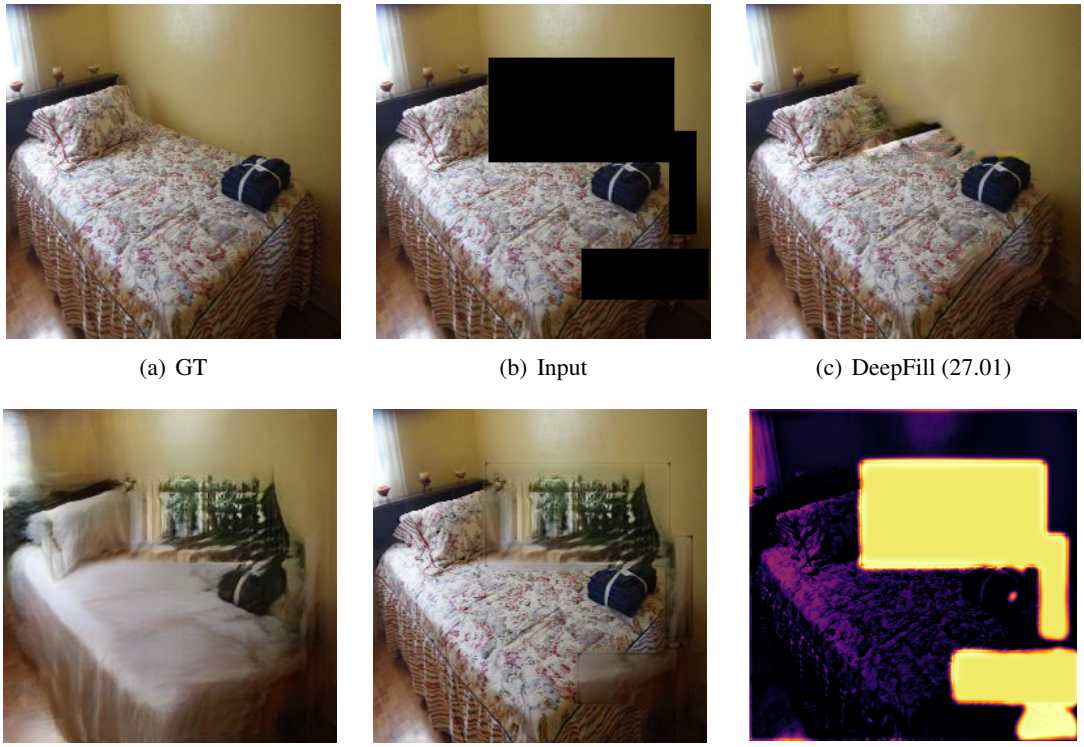
(a) GT (b) Input ($\sigma = 10$) (c) DnCNN (29.70)

(d) mGAN (19.86) (e) Ours (26.32) (f) $\phi$ (0.52)

(g) GT (h) Input ($\sigma = 6$) (i) DnCNN (34.09)

(j) mGAN (25.89) (k) Ours (31.85) (l) $\phi$ (0.66)

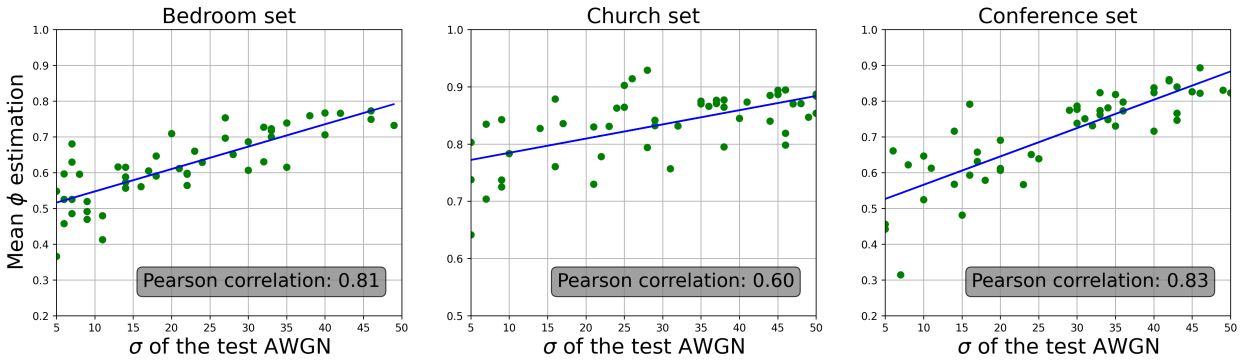
Figure 4.5 – Failure cases of AWGN removal. The quality of the generative-network inversion, which remains a very challenging task, is detrimental to our final results. Although our results significantly improve on the prior by exploiting the input observation by using our fusion weight, they still fall short of the task-specific DnCNN denoiser's results.



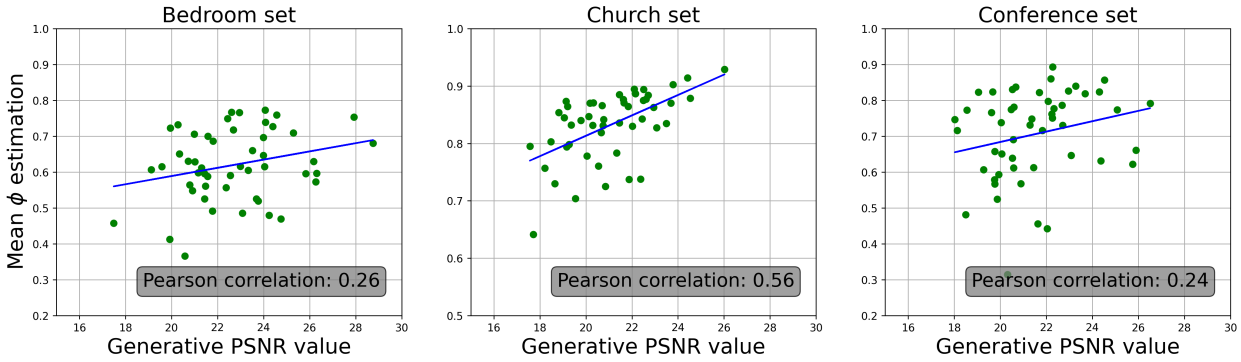
(a) GT (b) Input (c) DeepFill (27.01)

(d) mGAN (18.91) (e) Ours (20.26) (f) $\phi$ (0.33)

Figure 4.6 – Failure case in a randomized inpainting experiment. We note a misprediction in the $\phi$ mask in (f), in the bottom right corner. Our network mistakenly assumed the very dark region was a masked region. Note that in inpainting, data fidelity cannot be used in the masked area, and our results become directly dependent on the quality of the prior that, in this example, is not high.




(a) Correlation between $\phi$ and the AWGN $\sigma$




(b) Correlation between $\phi$ and the generative PSNR

Figure 4.7 – (a) Shows across three datasets the relation between the AWGN standard deviation in test images, which is directly related to signal quality, and our corresponding mean estimations for $\phi$. (b) Shows the same analysis but with respect to the PSNR of the generative network inversion results, which is directly related to the fitness of the prior. The results show a strong Pearson correlation factor between $\phi$ and signal quality (a), with the remaining factor of variation explained by the fitness of the prior (b) (e.g., Church set).

# Chapter 5

# Conclusion

## 5.1 Summary

In this thesis, we have studied the relations between data fidelity and learned image priors, for deep-learning-based image-restoration methods. Image restoration is centered around an interplay between these two terms, and although deep learning has improved quantitative restoration quality, it has come at the expense of the understanding of the underlying mechanisms, their generalization, and the interpretability and control over their hallucinations. We have analyzed, in Chapter 2, the behavior of deep learning SR and denoising methods in the frequency domain. We have noted a frequency-conditional learning, which is prone to degradation-model overfitting, for hallucinating missing components. We have designed a stochastic masking method, based on these findings, to regularize and improve the conditional learning of these restoration networks. In Chapter 3, we have investigated the optimality of a deep neural network on the fundamental denoising task, through a theoretically designed experimental setup. We observe that the network approaches a statistically optimal solution over its training range but fails to generalize. We have extended our solution, derived in our controlled experiment, to apply it to more general real images. By integrating internal learning of the noise level, and assuming a pixel-wise Gaussian prior on the underlying pixel distribution, we have shown improved real-image denoising results within a more interpretable architecture. This method however assumed a pixel-wise prior, and we have generalized it in Chapter 4 to full image priors that are learned using a generative network. We have presented a restoration framework that decouples the data fidelity information and the prior information that is extracted by projecting onto a generative network's space through regularized network inversion. These two terms are then fused by an adaptive learned module that adjusts to the observed data quality, as well as to the fitness of the prior to the input test image. This was supported by a correlation study over the relevant variation factors across multiple datasets. Our fusion weight provides spatially a structurally valid assessment of the amount of hallucination, which can be beneficial for user interpretability and potentially for downstream tasks. The framework we have presented also forms a theoretical generalization of a variety of classic restoration methods from the Bayesian restoration of Chapter 3 to dictionary-based restoration methods. We discuss, in the following section, future research directions related to each of the thesis chapters.

## 5.2 Future Research

We discuss in this section some future research directions that are closely connected to the chapters presented in this thesis.

### 5.2.1 Frequency Learning in Image Restoration

In Chapter 2, we noted how restoration networks can learn to add the missing frequencies in the degraded input image, but do so with little signal adaptation. With SFM, the networks become more aware of the actual content and degradation of the input image, rather than overfitting the training settings. This enables the networks to achieve better reconstruction by not adding frequency components that are already present in the input image, and also by adaptively adding more bands when those are missing.

If this reconstruction is disentangled per frequency band, every frequency band being restored separately by a dedicated sub-module, some bands would be reconstructed with a higher confidence, some with a lower confidence. At least in SR and in denoising, the higher the absolute frequency value is, the lower the reconstruction confidence would be. Furthermore, the more frequencies are lost in the degraded image, the lower the reconstruction confidence is.

A future research direction would be the design of a restoration approach that reconstructs images by reconstructing frequency components band by band, in a discretized or continuous way. Along with this reconstruction, a confidence curve could be implemented, as a function of spatial frequency. Such an approach would

- provide the user with more interpretable results through the disentangled component reconstruction, and the corresponding confidence values,
- and enable the user to select a variable reliability threshold, thus determining how much high frequency is desired to be added or how low of a confidence could be afforded and tweaking the final reconstruction accordingly.

This setup can be implemented over the DCT/Fourier domain, learned dictionaries, or more user-interpretable domains where basis vectors would be interpretable textures. Biomedical images with relatively constrained or limited texture patterns [162] would be a good test bed for developing such a method, and are also an important application where a control over non-confident hallucinations is crucial.

### 5.2.2 Estimation Theory Integration

We have designed, in Chapter 3, a theoretical setup tightly connected with additive denoising, where the image prior was pre-determined, hence leading us to having a closed-form optimal solution and enabling us to investigate the network's performance on this controlled experimental setup. In turn, we have extended this theoretical solution to a real-image denoising scenario with an observable improvement in the final denoising results. This was in part due to the explicit and disentangled learning of the noise level in the network's architecture.

This design approach can be used in other machine-learning-based tasks. Although it has recently become more common to add more explicit or disentangled internal learning, the use of theoretically defined experiments to guide novel architectures is still limited. It could be used to inspire more optimal methods, first under the theoretical setup, and then to extend them to real applications. This could also enable further insight into the inner workings that lead to the final solution.

Future research could explore tackling other imaging, computer vision, etc., problems by modeling the stochasticity of the ultimate solution of this problem with numerically tractable distributions that are as close to reality as possible. With this model taken as a given theoretical assumption, the solution could then be derived and its different terms integrated into the learning pipeline. This would

- provide more interpretable results through the intermediate terms,
- and reduce overfitting by enforcing certain theoretical rules and by the indirect regularization obtained from the learning of the different sub-tasks.

A more general, and more vague, research direction would ultimately be an optimal mixture between the rigid estimation theory solution, which is constrained by its theoretical assumptions, and the flexibility of data-driven empirical learning solutions. The integration between the two would need to be supervised by a meta-learned module that determines when to *follow reason* and when to *follow experience*, by understanding the current application's needs, as well as the strengths and shortcomings of each method. We began to touch on this direction in Chapter 4, where (1) our *reason* part was a simple bijective restoration step to completely preserve data fidelity, and (2) our supervising module for the mixture of the two approaches learned a pixel-level fusion of the two solutions.

### 5.2.3 Restoration with Decoupled Hallucination

In Chapter 4, our proposed method structurally provides a per-pixel map for fusing the prior information with the data-fidelity term. Future research can exploit this information, in addition to the restoration results, for downstream applications that currently take only the restored image as input. By training downstream methods to be aware of the faithfulness of the restoration result, the final results could be made more robust if not also more accurate. Our map would also enable us to train a judge that is able to disregard, or raise an alert, when restored input images have a too large $\phi$ map. This would either indicate that an image was severely degraded or that it contains significant hallucination in its restored version, both cases being prone to induce downstream errors.

Another research direction would be to extend our $\phi$ learning to classic methods. As discussed in Chapter 4, although some classic methods account for the quality of the signal, for example, the noise level, they do not account for the quality or fitness of the prior to the input image. A prior that fits rural images might fit differently and less accurately, for instance, urban images or biomedical images. Therefore, our adaptive approach could benefit classic methods, by learning the suitable weight to assign to the prior, according to its fitness to the input image, and also adapt in relation to the quality of the input data itself.

We have defined, in Chapter 4, both the fusion and its weight to be carried out over the spatial domain. This provides a visually interpretable map that spatially disentangles prior and data fidelity across pixels. It

would also be possible to define the hallucination in other domains, for instance, in the frequency domain. Such an approach could be designed by simply performing the fusion in the frequency domain or by transforming the entire pipeline into the frequency domain. On one hand, the former approach would transform the data-fidelity and the prior terms to the frequency domain and learn to predict a fusion weight map directly for the frequency domain. On the other hand, the latter approach would already transform the *input* to the frequency domain, thus relying on frequency-domain-based generative networks and performing the fusion in the frequency domain. Although this provides less interpretable fusion maps for users and for downstream tasks designed in the spatial domain, it can be useful for downstream applications designed in the frequency domain, and it would more directly correspond with the frequency-domain hallucination that we analyzed in Chapter 2.

# Bibliography


[1] Agustsson, E., Timofte, R.: NTIRE 2017 challenge on single image super-resolution: Dataset and study. In: Computer Vision and Pattern Recognition (CVPR) Workshops (2017)

[2] Aharon, M., Elad, M., Bruckstein, A.: K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on Signal Processing **54**(11), 4311–4322 (2006)

[3] Ahmed, N., Natarajan, T., Rao, K.R.: Discrete cosine transform. IEEE Transactions on Computers **100**(1), 90–93 (1974)

[4] Albright, M., McCloskey, S.: Source generator attribution via inversion. In: Computer Vision and Pattern Recognition (CVPR) Workshops (2019)

[5] Amano, T., Unal, C.T., Paré, D.: Synaptic correlates of fear extinction in the amygdala. Nature Neuroscience **13**(4), 489 (2010)

[6] Anwar, S., Barnes, N.: Real image denoising with feature attention. International Conference on Computer Vision (ICCV) (2019)

[7] Anwar, S., Huynh, C.P., Porikli, F.: Image deblurring with a class-specific prior. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(9), 2112–2130 (2018)

[8] Bao, J., Chen, D., Wen, F., Li, H., Hua, G.: CVAE-GAN: fine-grained image generation through asymmetric training. In: International Conference on Computer Vision (ICCV). pp. 2745–2754 (2017)

[9] Batson, J., Royer, L.: Noise2Self: Blind denoising by self-supervision. In: International Conference on Machine Learning (ICML) (2019)

[10] Bau, D., Strobelt, H., Peebles, W., Zhou, B., Zhu, J.Y., Torralba, A., et al.: Semantic photo manipulation with a generative image prior. arXiv preprint arXiv:2005.07727 (2020)

[11] Beckouche, S., Starck, J.L., Fadili, J.: Astronomical image denoising using dictionary learning. Astronomy & Astrophysics **556**, A132 (2013)

[12] Benazza-Benyahia, A., Pesquet, J.C.: Building robust wavelet estimators for multicomponent images using Stein's principle. IEEE Transactions on Image Processing **14**(11), 1814–1830 (2005)

[13] Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(8), 1798–1828 (2013)

[14] Brooks, T., Mildenhall, B., Xue, T., Chen, J., Sharlet, D., Barron, J.T.: Unprocessing images for learned raw denoising. In: Computer Vision and Pattern Recognition (CVPR). pp. 11036–11045 (2019)

[15] Buades, A., Coll, B., Morel, J.M.: A non-local algorithm for image denoising. In: Computer Vision and Pattern Recognition (CVPR). pp. 60–65 (2005)

[16] Buades, A., Coll, B., Morel, J.M.: Nonlocal image and movie denoising. IJCV **76**(2), 123–139 (2008)

[17] Burger, H.C., Schuler, C.J., Harmeling, S.: Image denoising: Can plain neural networks compete with BM3D? In: Computer Vision and Pattern Recognition (CVPR). pp. 2392–2399 (2012)

[18] Burton, G.J., Moorhead, I.R.: Color and spatial structure in natural scenes. Applied Optics **26**(1), 157–170 (1987)

[19] Cai, J., Zeng, H., Yong, H., Cao, Z., Zhang, L.: Toward real-world single image super-resolution: A new benchmark and a new model. In: International Conference on Computer Vision (ICCV) (2019)

[20] Chan, T.M., Zhang, J., Pu, J., Huang, H.: Neighbor embedding based super-resolution algorithm through edge detection and feature selection. Pattern Recognition Letters (2009)

[21] Chatterjee, P., Joshi, N., Kang, S.B., Matsushita, Y.: Noise suppression in low-light images through joint denoising and demosaicing. In: Computer Vision and Pattern Recognition (CVPR) (2011)

[22] Chen, J., Chen, J., Chao, H., Yang, M.: Image blind denoising with generative adversarial network based noise modeling. In: Computer Vision and Pattern Recognition (CVPR). pp. 3155–3164 (2018)

[23] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In: Neural Information Processing Systems (NeurIPS). pp. 2172–2180 (2016)

[24] Chen, Y., Pock, T.: Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence **39**(6), 1256–1272 (2016)

[25] Chen, Y., Yu, W., Pock, T.: On learning optimized reaction diffusion processes for effective image restoration. In: Computer Vision and Pattern Recognition (CVPR). pp. 5261–5269 (2015)

[26] Cho, T.S., Joshi, N., Zitnick, C.L., Kang, S.B., Szeliski, R., Freeman, W.T.: A content-aware image prior. In: Computer Vision and Pattern Recognition (CVPR). pp. 169–176 (2010)

[27] Choi, S., Isidoro, J., Getreuer, P., Milanfar, P.: Fast, trainable, multiscale denoising. In: International Conference on Image Processing (ICIP). pp. 963–967 (2018)

[28] Cohen, R., Elad, M., Milanfar, P.: Regularization by denoising via fixed-point projection (red-pro). arXiv preprint arXiv:2008.00226 (2020)

[29] Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Transactions on Image Processing **16**(8), 2080–2095 (2007)

[30] Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: BM3D image denoising with shape-adaptive principal component analysis. In: Signal Processing with Adaptive Sparse Structured Representations (2009)

[31] Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.O.: Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space. In: International Conference on Image Processing (ICIP). pp. 313–316 (2007)

[32] Descartes, R.: Meditations on first philosophy: With selections from the objections and replies. Oxford University Press (2008)

[33] Deshpande, A., Rock, J., Forsyth, D.: Learning large-scale automatic image colorization. In: International Conference on Computer Vision (ICCV). pp. 567–575 (2015)

[34] Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. In: International Conference on Learning Representations (ICLR) (2017)

[35] Dong, W., Shi, G., Li, X., Ma, Y., Huang, F.: Compressive sensing via nonlocal low-rank regularization. IEEE Transactions on Image Processing **23**(8), 3618–3632 (2014)

[36] Dong, W., Zhang, L., Shi, G.: Centralized sparse representation for image restoration. In: International Conference on Computer Vision (ICCV). pp. 1259–1266 (2011)

[37] Dong, W., Zhang, L., Shi, G., Li, X.: Nonlocally centralized sparse representation for image restoration. IEEE Transactions on Image Processing **22**(4), 1620–1630 (2013)

[38] Donoho, D.L.: De-noising by soft-thresholding. IEEE Transactions on Information Theory **41**(3), 613–627 (1995)

[39] Efrat, N., Glasner, D., Apartsin, A., Nadler, B., Levin, A.: Accurate blur models vs. image priors in single image super-resolution. In: International Conference on Computer Vision (ICCV) (2013)

[40] El Helou, M., Dümbgen, F., Süsstrunk, S.: AAM: An assessment metric of axial chromatic aberration. In: International Conference on Image Processing (ICIP) (2018)

[41] El Helou, M., Dümbgen, F., Süsstrunk, S.: AL2: Progressive activation loss for learning general representations in classification neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 4007–4011 (2020)

[42] El Helou, M., Mandt, S., Krause, A., Beardsley, P.: Mobile robotic painting of texture. In: International Conference on Robotics and Automation (ICRA) (2019)

[43] El Helou, M., Sadeghipoor, Z., Süsstrunk, S.: Correlation-based deblurring leveraging multispectral chromatic aberration in color and near-infrared joint acquisition. In: International Conference on Image Processing (ICIP). pp. 1402–1406 (2017)

[44] El Helou, M., Süsstrunk, S.: BIGPrior: Towards decoupling learned prior hallucination and data fidelity in image restoration. arXiv preprint arXiv:2011.01406 (2020)

[45] El Helou, M., Süsstrunk, S.: Blind universal Bayesian image denoising with Gaussian noise level learning. IEEE Transactions on Image Processing **29**, 4885–4897 (2020)

[46] El Helou, M., Zhou, R., Süsstrunk, S.: Stochastic frequency masking to improve super-resolution and denoising networks. In: European Conference on Computer Vision (ECCV) (2020)

[47] Elad, M.: Image Denoising, pp. 273–307. Springer New York, New York, NY (2010)

[48] Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image Processing **15**(12), 3736–3745 (2006)

[49] Field, D.J.: Relations between the statistics of natural images and the response properties of cortical cells. Journal of the Optical Society of America (JOSA) **4**(12), 2379–2394 (1987)

[50] Foi, A., Trimeche, M., Katkovnik, V., Egiazarian, K.: Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data. IEEE Transactions on Image Processing **17**(10), 1737–1754 (2008)

[51] Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-based super-resolution. Computer Graphics and Applications (2002)

[52] Giryes, R., Elad, M.: Sparsity-based Poisson denoising with dictionary learning. IEEE Transactions on Image Processing **23**(12), 5057–5069 (2014)

[53] Godard, C., Matzen, K., Uyttendaele, M.: Deep burst denoising. In: European Conference on Computer Vision (ECCV). pp. 538–554 (2018)

[54] Gu, J., Lu, H., Zuo, W.Z., Dong, C.: Blind super-resolution with iterative kernel correction. In: Computer Vision and Pattern Recognition (CVPR) (2019)

[55] Gu, J., Shen, Y., Zhou, B.: Image processing using multi-code GAN prior. In: Computer Vision and Pattern Recognition (CVPR). pp. 3012–3021 (2020)

[56] Gu, S., Zhang, L., Zuo, W., Feng, X.: Weighted nuclear norm minimization with application to image denoising. In: Computer Vision and Pattern Recognition (CVPR) (2014)

[57] Gunturk, B.K., Batur, A.U., Altunbasak, Y., Hayes, M.H., Mersereau, R.M.: Eigenface-domain super-resolution for face recognition. IEEE Transactions on Image Processing **12**(5), 597–606 (2003)

[58] Hasan, M., El-Sakka, M.R.: Improved BM3D image denoising using SSIM-optimized Wiener filter. EURASIP Journal on Image and Video Processing **2018**(1), 25 (2018)

[59] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016)

[60] Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Computer Vision and Pattern Recognition (CVPR). pp. 5197–5206 (2015)

[61] Hume, D.: An enquiry concerning human understanding: A critical edition, vol. 3. Oxford University Press (2000)

[62] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning (ICML). pp. 448–456 (2015)

[63] Jain, V., Seung, S.: Natural image denoising with convolutional networks. In: Neural Information Processing Systems (NeurIPS). pp. 769–776 (2009)

[64] Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision (ECCV) (2016)

[65] Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (ICLR) (2018)

[66] Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Computer Vision and Pattern Recognition (CVPR). pp. 4401–4410 (2019)

[67] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Computer Vision and Pattern Recognition (CVPR). pp. 8110–8119 (2020)

[68] Khan, S.H., Hayat, M., Porikli, F.: Regularization of deep neural networks with spectral dropout. Neural Networks **110**, 82–90 (2019)

[69] Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: Computer Vision and Pattern Recognition (CVPR) (2016)

[70] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[71] Krishnan, D., Fergus, R.: Fast image deconvolution using hyper-Laplacian priors. In: Neural Information Processing Systems (NeurIPS). pp. 1033–1041 (2009)

[72] Krull, A., Buchholz, T.O., Jug, F.: Noise2Void-learning denoising from single noisy images. In: Computer Vision and Pattern Recognition (CVPR) (2019)

[73] Laplace, P.S.: Pierre-Simon Laplace Philosophical Essay on Probabilities: Translated from the fifth French edition of 1825 With Notes by the Translator, vol. 13. Springer Science & Business Media (1998)

[74] Le Montagner, Y., Angelini, E.D., Olivo-Marin, J.C.: An unbiased risk estimator for image denoising in the presence of mixed Poisson–Gaussian noise. IEEE Transactions on Image Processing **23**(3), 1255–1268 (2014)

[75] Lebrun, M., Buades, A., Morel, J.M.: A nonlocal Bayesian image denoising algorithm. SIAM Journal on Imaging Sciences **6**(3), 1665–1688 (2013)

[76] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: Computer Vision and Pattern Recognition (CVPR) (2017)

[77] Lee, S., Cho, D., Kim, J., Kim, T.H.: Self-supervised fast adaptation for denoising via meta-learning. arXiv preprint arXiv:2001.02899 (2020)

[78] Lefkimmiatis, S.: Non-local color image denoising with convolutional neural networks. In: Computer Vision and Pattern Recognition (CVPR). pp. 3587–3596 (2017)

[79] Lefkimmiatis, S.: Universal denoising networks: A novel CNN architecture for image denoising. In: Computer Vision and Pattern Recognition (CVPR). pp. 3204–3213 (2018)

[80] Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., Aila, T.: Noise2Noise: Learning image restoration without clean data. In: International Conference on Machine Learning (ICML) (2018)

[81] Li, S., Yin, H., Fang, L.: Group-sparse representation with dictionary learning for medical image denoising and fusion. IEEE Transactions on Biomedical Engineering **59**(12), 3450–3459 (2012)

[82] Liu, D., Wen, B., Liu, X., Wang, Z., Huang, T.S.: When image denoising meets high-level vision tasks: a deep learning approach. In: International Joint Conference on Artificial Intelligence (IJCAI). pp. 842–848 (2018)

[83] Liu, Y.C., Yeh, Y.Y., Fu, T.C., Wang, S.D., Chiu, W.C., Frank Wang, Y.C.: Detach and adapt: Learning cross-domain disentangled deep representation. In: Computer Vision and Pattern Recognition (CVPR). pp. 8867–8876 (2018)

[84] Lobas, M.A., Tao, R., Nagai, J., Kronschläger, M.T., Borden, P.M., Marvin, J.S., Looger, L.L., Khakh, B.S.: A genetically encoded single-wavelength sensor for imaging cytosolic and cell surface ATP. Nature Communications **10**(1), 711 (2019)

[85] Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. In: International Conference on Learning Representations (ICLR) (2017)

[86] Luisier, F., Blu, T., Unser, M.: Image denoising in mixed Poisson-Gaussian noise. IEEE Transactions on Image Processing **20**(3), 696–708 (2011)

[87] Mahmood, F., Toots, M., Öfverstedt, L.G., Skoglund, U.: 2D discrete Fourier transform with simultaneous edge artifact removal for real-time applications. In: IEEE International Conference on Field Programmable Technology. pp. 236–239 (2015)

[88] Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Non-local sparse models for image restoration. In: International Conference on Computer Vision (ICCV). pp. 2272–2279 (2009)

[89] Makitalo, M., Foi, A.: Optimal inversion of the generalized Anscombe transformation for Poisson-Gaussian noise. IEEE Transactions on Image Processing **22**(1), 91–103 (2012)

[90] Makitalo, M., Foi, A.: Noise parameter mismatch in variance stabilization, with an application to Poisson–Gaussian noise estimation. IEEE Transactions on Image Processing **23**(12), 5348–5359 (2014)

[91] Martin, D., Fowlkes, C., Tal, D., Malik, J., et al.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: International Conference on Computer Vision (ICCV). pp. 416–423 (2001)

[92] Matsui, Y., Ito, K., Aramaki, Y., Fujimoto, A., Ogawa, T., Yamasaki, T., Aizawa, K.: Sketch-based manga retrieval using manga109 dataset. Multimedia Tools and Applications **76**(20), 21811–21838 (2017)

[93] Menon, S., Damian, A., Hu, S., Ravi, N., Rudin, C.: PULSE: Self-supervised photo upsampling via latent space exploration of generative models. In: Computer Vision and Pattern Recognition (CVPR). pp. 2437–2445 (2020)

[94] Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: International Conference on Machine Learning (ICML). pp. 807–814 (2010)

[95] Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An iterative regularization method for total variation-based image restoration. Multiscale Modeling & Simulation **4**(2), 460–489 (2005)

[96] Pan, X., Zhan, X., Dai, B., Lin, D., Loy, C.C., Luo, P.: Exploiting deep generative prior for versatile image restoration and manipulation. In: European Conference on Computer Vision (ECCV) (2020)

[97] Pangle, T.L.: The laws of Plato. University of Chicago Press (1988)

[98] Parikh, N., Boyd, S., et al.: Proximal algorithms. Foundations and Trends® in Optimization **1**(3), 127–239 (2014)

[99] Park, D.S., Chan, W., Zhang, Y., Chiu, C.C., Zoph, B., Cubuk, E.D., Le, Q.V.: SpecAugment: A simple data augmentation method for automatic speech recognition. arXiv preprint arXiv:1904.08779 (2019)

[100] Peled, S., Yeshurun, Y.: Superresolution in MRI: application to human white matter fiber tract visualization by diffusion tensor imaging. Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine **45**(1), 29–35 (2001)

[101] Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence **12**(7), 629–639 (1990)

[102] Plötz, T., Roth, S.: Benchmarking denoising algorithms with real photographs. In: Computer Vision and Pattern Recognition (CVPR). pp. 2750–2759 (2017)

[103] Plötz, T., Roth, S.: Neural nearest neighbors networks. In: Neural Information Processing Systems (NeurIPS) (2018)

[104] Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)

[105] Remez, T., Litany, O., Giryes, R., Bronstein, A.M.: Class-aware fully convolutional Gaussian and Poisson denoising. IEEE Transactions on Image Processing **27**(11), 5707–5722 (2018)

[106] Romdhani, S., Vetter, T.: Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In: Computer Vision and Pattern Recognition (CVPR). pp. 986–993 (2005)

[107] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI). pp. 234–241 (2015)

[108] Roth, S., Black, M.J.: Fields of experts. International Journal of Computer Vision (IJCV) **82**(2), 205 (2009)

[109] Rubinstein, R., Peleg, T., Elad, M.: Analysis K-SVD: A dictionary-learning algorithm for the analysis sparse model. IEEE Transactions on Signal Processing **61**(3), 661–677 (2012)

[110] Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena **60**(1-4), 259–268 (1992)

[111] Sardy, S., Tseng, P., Bruce, A.: Robust wavelet denoising. IEEE Transactions on Signal Processing **49**(6), 1146–1152 (2001)

[112] Schmidt, U., Roth, S.: Shrinkage fields for effective image restoration. In: Computer Vision and Pattern Recognition (CVPR). pp. 2774–2781 (2014)

[113] Schulter, S., Leistner, C., Bischof, H.: Fast and accurate image upscaling with super-resolution forests. In: Computer Vision and Pattern Recognition (CVPR) (2015)

[114] Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Computer Vision and Pattern Recognition (CVPR) (2016)

[115] Shi, W., Caballero, J., Ledig, C., Zhuang, X., Bai, W., Bhatia, K., de Marvao, A.M.S.M., Dawes, T., O'Regan, D., Rueckert, D.: Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. In: Medical Image Computing and Computer-Assisted Intervention (MICCAI). pp. 9–16 (2013)

[116] Shocher, A., Cohen, N., Irani, M.: "zero-shot" super-resolution using deep internal learning. In: Computer Vision and Pattern Recognition (CVPR) (2018)

[117] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations (ICLR) (2015)

[118] Sindagi, V.A., Oza, P., Yasarla, R., Patel, V.M.: Prior-based domain adaptive object detection for hazy and rainy conditions. In: European Conference on Computer Vision (ECCV) (2020)

[119] Starck, J.L., Candès, E.J., Donoho, D.L.: The curvelet transform for image denoising. IEEE Transactions on Image Processing **11**, 670–684 (2002)

[120] Strang, G.: The discrete cosine transform. SIAM review **41**(1), 135–147 (1999)

[121] Suganuma, M., Ozay, M., Okatani, T.: Exploiting the potential of standard convolutional autoencoders for image restoration by evolutionary search. In: International Conference on Machine Learning (ICML). pp. 4778–4787 (2018)

[122] Sun, J., Xu, Z., Shum, H.Y.: Image super-resolution using gradient profile prior. In: Computer Vision and Pattern Recognition (CVPR) (2008)

[123] Sun, L., Hays, J.: Super-resolution from internet-scale scene matching. In: International Conference on Computational Photography (ICCP). pp. 1–12. IEEE (2012)

[124] Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A.A., Hardt, M.: Test-time training with self-supervision for generalization under distribution shifts. In: International Conference on Machine Learning (ICML) (2020)

[125] Tai, Y., Yang, J., Liu, X., Xu, C.: MemNet: A persistent memory network for image restoration. In: International Conference on Computer Vision (ICCV) (2017)

[126] Tatarchenko, M., Richter, S.R., Ranftl, R., Li, Z., Koltun, V., Brox, T.: What do single-view 3D reconstruction networks learn? In: Computer Vision and Pattern Recognition (CVPR). pp. 3405–3414 (2019)

[127] Thornton, M.W., Atkinson, P.M., Holland, D.: Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping. International Journal of Remote Sensing (IJRS) **27**(3), 473–491 (2006)

[128] Tian, C., Xu, Y., Fei, L., Yan, K.: Deep learning for image denoising: A survey. arXiv preprint arXiv:1810.05052 (2018)

[129] Tolhurst, D., Tadmor, Y., Chao, T.: Amplitude spectra of natural images. Ophthalmic and Physiological Optics **12**(2), 229–232 (1992)

[130] Torralba, A., Oliva, A.: Statistics of natural image categories. Network: Computation in Neural Systems **14**(3), 391–412 (2003)

[131] Ulyanov, D., Vedaldi, A., Lempitsky, V.: Deep image prior. In: Computer Vision and Pattern Recognition (CVPR). pp. 9446–9454 (2018)

[132] Wallace, G.K.: The JPEG still picture compression standard. IEEE Transactions on Consumer Electronics **38**(1) (1992)

[133] Wang, G., Lopez-Molina, C., De Baets, B.: Blob reconstruction using unilateral second order Gaussian kernels with application to high-ISO long-exposure image denoising. In: International Conference on Computer Vision (ICCV). pp. 4817–4825 (2017)

[134] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., Loy, C.C.: ESRGAN: Enhanced super-resolution generative adversarial networks. In: European Conference on Computer Vision (ECCV) Workshops (2018)

[135] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004)

[136] Weiss, Y., Freeman, W.T.: What makes a good model of natural images? In: Computer Vision and Pattern Recognition (CVPR). pp. 1–8 (2007)

[137] Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: Neural Information Processing Systems (NeurIPS). pp. 341–349 (2012)

[138] Xu, J., Osher, S.: Iterative regularization and nonlinear inverse scale space applied to wavelet-based denoising. IEEE Transactions on Image Processing **16**(2), 534–544 (2007)

[139] Xu, J., Zhang, L., Zuo, W., Zhang, D., Feng, X.: Patch group based nonlocal self-similarity prior learning for image denoising. In: International Conference on Computer Vision (ICCV). pp. 244–252 (2015)

[140] Yang, C.Y., Ma, C., Yang, M.H.: Single-image super-resolution: A benchmark. In: European Conference on Computer Vision (ECCV) (2014)

[141] Yang, J., Kannan, A., Batra, D., Parikh, D.: LR-GAN: Layered recursive generative adversarial networks for image generation. In: International Conference on Learning Representations (ICLR) (2017)

[142] Yi, Z., Zhang, H., Tan, P., Gong, M.: Dualgan: Unsupervised dual learning for image-to-image translation. In: International Conference on Computer Vision (ICCV). pp. 2849–2857 (2017)

[143] Yongcheng, J., Yezhou, Y., Zunlei, F., Jingwen, Y., Yizhou, Y., Song, M.: Neural style transfer: A review. arXiv preprint arXiv:1705.04058v6 (2018)

[144] Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., Xiao, J.: LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)

[145] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: Computer Vision and Pattern Recognition (CVPR). pp. 5505–5514 (2018)

[146] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: International Conference on Computer Vision (ICCV). pp. 4471–4480 (2019)

[147] Zagoruyko, S., Komodakis, N.: Wide residual networks. In: British Machine Vision Conference (BMVC) (2016)

[148] Zamir, A.R., Sax, A., Shen, W., Guibas, L.J., Malik, J., Savarese, S.: Taskonomy: Disentangling task transfer learning. In: Computer Vision and Pattern Recognition (CVPR). pp. 3712–3722 (2018)

[149] Zhang, J., Li, W., Ogunbona, P.: Joint geometrical and statistical alignment for visual domain adaptation. In: Computer Vision and Pattern Recognition (CVPR). pp. 1859–1867 (2017)

[150] Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. IEEE Transactions on Image Processing **26**(7), 3142–3155 (2017)

[151] Zhang, K., Zuo, W., Gu, S., Zhang, L.: Learning deep CNN denoiser prior for image restoration. In: Computer Vision and Pattern Recognition (CVPR). pp. 3929–3938 (2017)

[152] Zhang, K., Zuo, W., Zhang, L.: FFDNet: Toward a fast and flexible solution for CNN-based image denoising. IEEE Transactions on Image Processing **27**(9), 4608–4622 (2018)

[153] Zhang, K., Zuo, W., Zhang, L.: Learning a single convolutional super-resolution network for multiple degradations. In: Computer Vision and Pattern Recognition (CVPR). pp. 3262–3271 (2018)

[154] Zhang, K., Zuo, W., Zhang, L.: Deep plug-and-play super-resolution for arbitrary blur kernels. In: Computer Vision and Pattern Recognition (CVPR) (2019)

[155] Zhang, R., Isola, P., Efros, A.: Colorful image colorization. In: European Conference on Computer Vision (ECCV). pp. 649–666 (2016)

[156] Zhang, R., Isola, P., Efros, A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Computer Vision and Pattern Recognition (CVPR). pp. 586–595 (2018)

[157] Zhang, X., Chen, Q., Ng, R., Koltu, V.: Zoom to learn, learn to zoom. In: International Conference on Computer Vision (ICCV) (2019)

[158] Zhang, Y., Zhu, Y., Nichols, E., Wang, Q., Zhang, S., Smith, C., Howard, S.: A Poisson-Gaussian denoising dataset with real fluorescence microscopy images. In: Computer Vision and Pattern Recognition (CVPR) (2019)

[159] Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: European Conference on Computer Vision (ECCV) (2018)

[160] Zhang, Y., Li, K., Li, K., Zhong, B., Fu, Y.: Residual non-local attention networks for image restoration. In: International Conference on Learning Representations (ICLR) (2019)

[161] Zhang, Y., Guo, Y., Jin, Y., Luo, Y., He, Z., Lee, H.: Unsupervised discovery of object landmarks as structural representations. In: Computer Vision and Pattern Recognition (CVPR). pp. 2694–2703 (2018)

[162] Zhou, R., El Helou, M., Sage, D., Laroche, T., Seitz, A., Süsstrunk, S.: W2S: Microscopy data with joint denoising and super-resolution for widefield to SIM mapping. In: European Conference on Computer Vision (ECCV) Workshops (2020)

[163] Zhou, R., Lahoud, F., El Helou, M., Süsstrunk, S.: A comparative study on wavelets and residuals in deep super resolution. In: Electronic Imaging (2019)

[164] Zhou, R., Süsstrunk, S.: Kernel modeling super-resolution on real low-resolution images. In: International Conference on Computer Vision (ICCV) (2019)

[165] Zoran, D., Weiss, Y.: From learning models of natural image patches to whole image restoration. In: International Conference on Computer Vision (ICCV) (2011)

# CV

## Contact - Majed EL HELOU

**Email:** majed.elhelou.1@gmail.com

## Education

Ph.D. in the Image and Visual Representation Lab (IVRL).
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.
**Research experience**: Signal/image processing, computational photography/imaging, applied machine learning, deep learning, stochastic estimation theory, large image optimization, and computer vision.
**Publications** are listed on **https://majedelhelou.github.io/publications**
**Code** repositories can be found on **https://github.com/majedelhelou**

## Publications

- **M. El Helou** and S. Süsstrunk, "BIGPrior: Towards Decoupling Learned Prior Hallucination and Data Fidelity in Image Restoration," *arXiv preprint arXiv:2011.01406*, 2020.

- **M. El Helou***, R. Zhou* and S. Süsstrunk, "Stochastic Frequency Masking to Improve Super-Resolution and Denoising Networks," *in European Conference on Computer Vision (ECCV)*, 2020. (*similar contribution)

- R. Zhou*, **M. El Helou***, D. Sage, T. Laroche, A. Seitz and S. Süsstrunk, "W2S: Microscopy Data with Joint Denoising and Super-Resolution for Widefield to SIM Mapping," *in European Conference on Computer Vision Workshops (ECCVW)*, 2020. (*similar contribution)

- **M. El Helou**, R. Zhou, S. Süsstrunk, R. Timofte, et al., "AIM 2020: Scene Relighting and Illumination Estimation Challenge," *in European Conference on Computer Vision Workshops (ECCVW)*, 2020.

- **M. El Helou** and S. Süsstrunk, "Blind Universal Bayesian Image Denoising with Gaussian Noise Level Learning," *in IEEE Transactions on Image Processing (TIP)*, vol. 29, pp. 4885-4897, 2020.

**-** **M. El Helou**, R. Zhou, F. Schmutz, F. Guibert and S. Süsstrunk, “Divergence-Based Adaptive Extreme Video Completion,” *in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

**-** **M. El Helou**, F. Dümbgen and S. Süsstrunk, “AL2: Progressive Activation Loss for Learning General Representations in Classification Neural Networks,” *in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

**-** F. Dümbgen, **M. El Helou** and A. Scholefield, “Realizability of Planar Point Embeddings from Angle Measurements,” *in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

**-** **M. El Helou**, S. Mandt, A. Krause and P. Beardsley, “Mobile Robotic Painting of Texture,” *in IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

**-** **M. El Helou**, M. Shahpaski and S. Süsstrunk, “Solving the Depth Ambiguity in Single-Perspective Images,” *in Imaging Systems, Image Processing, and Displays (OSA Continuum)*, Vol. 2, No. 10, 2019.

**-** **M. El Helou**, M. Shahpaski and S. Süsstrunk, “Closed-Form Solution to Disambiguate Defocus Blur in Single-Perspective Images,” *in Mathematics in Imaging, Imaging and Applied Optics Congress, (OSA Math)*, 2019.

**-** R. Zhou, F. Lahoud, **M. El Helou** and S. Süsstrunk, “A Comparative Study on Wavelets and Residuals in Deep Super Resolution,” *in IS&T/SPIE Electronic Imaging*, 2019.

**-** **M. El Helou**, F. Dümbgen and S. Süsstrunk, "AAM: an Assessment Metric of Axial Chromatic Aberration," *in IEEE International Conference on Image Processing (ICIP)*, 2018.

**-** F. Dümbgen*, **M. El Helou***, N. Gucevska and S. Süsstrunk, "Near-Infrared Fusion for Photorealistic Image Dehazing," *in IS&T/SPIE Electronic Imaging*, 2018. (*equal contribution)

**-** **M. El Helou**, Z. Sadeghipoor and S. Süsstrunk, "Correlation-Based Deblurring Leveraging Multispectral Chromatic Aberration in Color and Near-Infrared Joint Acquisition," *in IEEE International Conference on Image Processing (ICIP)*, 2017.

## Technical tutorial

**-** **M. El Helou**, F. Dümbgen, R. Achanta and S. Süsstrunk, “Fourier-Domain Optimization for Image Processing,” *arXiv preprint arXiv:1809.04187*, 2018.