

NOISE2NOISEFLOW: REALISTIC CAMERA NOISE MODELING WITHOUT CLEAN IMAGES

ALI MALEKY

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

AUGUST, 2022

© ALI MALEKY, 2022

Abstract

Image noise modeling is a long-standing problem with many applications in computer vision. Early attempts that propose simple models, such as signal-independent additive white Gaussian noise or the heteroscedastic Gaussian noise model (a.k.a., camera noise level function) are not sufficient to learn the complex behavior of the camera sensor noise. Recently, more complex learning-based models have been proposed that yield better results in noise synthesis and downstream tasks, such as denoising. However, their dependence on supervised data (*i.e.*, paired clean images) is a limiting factor given the challenges in producing ground-truth images. This paper proposes a framework for training a noise model and a denoiser simultaneously while relying only on pairs of noisy images rather than noisy/clean paired image data. We apply this framework to the training of the Noise Flow architecture. The noise synthesis and density estimation results show that our framework outperforms previous signal-processing-based noise models and is on par with its supervised counterpart. The trained denoiser is also shown to significantly improve upon both supervised and weakly supervised baseline denoising approaches. The results indicate that the joint training of a denoiser and a noise model yields significant improvements in the denoiser.

Acknowledgements

I have been so lucky to have great individuals as my closest souls throughout my master's studies without whom I could not do so well.

I would like to thank my supervisors, Michael S. Brown, and Marcus A. Brubaker. As an international student living away from family, I have always seen them as my go-to for my hardest problems. They have always been so helpful and understanding and I truly see them as my father. I learned a lot from them in the past two years, both educationally and spiritually. I feel so lucky to have had them in my life and it is so sad for me to leave them.

I would also thank my spouse, Mohaddeseh as she has always been there for me. I felt her support when I needed it the most. She has always been willing to sacrifice for my good. I am genuinely grateful for what she gave me during times of ups and downs.

Despite living far away, my family has been very supportive. Thanks to the advances in technology, I have had the chance to connect to them at my will. They have always helped me the best way they could.

Lastly, I want to thank my friends and labmates for their kindness and help. I would specifically name Shayan Kousha, who was more than a friend. I constantly received his feedback and help during my project and helped me face challenges whenever needed.

The individuals mentioned above have helped me be who I am. I am and will continue to acknowledge their support.

Contents

Abstract	ii
Acknowledgements	iii
Contents	iv
List of Tables	vi
List of Figures	viii
Nomenclature	xi
1 Introduction	1
1.1 Contributions	5
2 Background and Related Work	7
2.1 Image Noise Modeling	8
2.2 Supervised Image Denoising	13
2.3 Noise Datasets	15
2.4 Image Denoising without Clean Images	17
3 Noise2NoiseFlow	21
3.1 R2RFlow	25

4 Experiments	28
4.1 Dataset	29
4.2 Metrics	30
4.3 Training Details	31
4.4 Results	31
4.4.1 Noise Modeling	32
4.4.2 Noise Reduction	33
4.4.3 Ablation Studies	36
4.5 Training with Individual Noisy Images	40
4.6 Synthetic Noise Experiment	41
4.7 Failure Cases	43
5 Conclusion	45
5.1 Broader Impact	48
5.2 Future Work	48
Bibliography	50

List of Tables

4.1	Negative log-likelihood per dimension and D_{KL} results on test data for baseline models and our proposed Noise2NoiseFlow model. Noise2NoiseFlow significantly improves over AWGN and Camera NLF and is on par with the Noise Flow model, while requiring no clean images. It also improves over separately training a Noise2Noise denoiser and NoiseFlow ($N2N+NF$), <i>demonstrating the value of joint training</i>	33
4.2	Per camera KL divergence performance of our model Noise2NoiseFlow compared to the baselines on three camera sensors for which the Calibrated P-G model is defined as well as the aggregate results on the test data for these three sensors.	33
4.3	Denoising results from a DnCNN trained with supervised noisy/clean paired data from SIDD-Medium, Noise2Noise with a DnCNN trained on noisy/noisy image pairs, a DnCNN trained with noise samples generated from a supervised Noise Flow model, and our denoiser model trained on the same noisy/noisy data measured by PSNR and SSIM on the test set.	36
4.4	Performance on test data from our ablated models. Each row corresponds to a specific choice for the noise model and the denoiser architecture. NLF corresponds to our modified version of the heteroscedastic Gaussian model implemented as a bijective normalizing flow transformation.	40

4.5	Performance on test data from our proposed R2RFlow formulation that requires only single noisy samples and no supervision in any forms compared to Noise2NoiseFlow (weak supervision) and the Noise Flow model (supervised).	41
-----	--	----

List of Figures

1.1	Overview of Noise2NoiseFlow. Given pairs of noisy images of the same scene, Noise2NoiseFlow simultaneously trains both a noise model and a denoiser.	3
1.2	Sample results from our Noise2NoiseFlow model. (top) Noise synthesis results from Camera NLF, Noise Flow, and Noise2NoiseFlow compared to the real noise in the SIDD dataset. Noise generated by Noise2NoiseFlow is the most similar to the real noise both visually and in KL divergence but without requiring clean, noise-free images. (bottom) Example denoising results from the jointly trained denoiser compared to its supervised DnCNN baseline, and a DnCNN trained with Noise2Noise loss.	4
2.1	An overview of the Noise Flow model proposed by Abdelhamed et al. [2]. The forward pass takes the residual image noise and transforms it into a known base distribution, which in this case is an isotropic Gaussian model. The transformation is conditioned on the clean image, sensor type, and gain factor (ISO level). The backward pass takes a sample from the isotropic Gaussian distribution and transforms it into a sample from the image noise distribution, conditioned on the clean image, sensor type, and sensor gain.	11
2.2	The overall architecture of the DnCNN [51] model. “BN” denotes Batch Normalization [18].	13

2.3	Overview of the U-Net architecture introduced by Ronneberger et al. [43]. The figure is re-produced from the original paper. Note that the architecture they introduce was originally used for biomedical image segmentation, and not image denoising.	14
2.4	A sample image pair from the Darmstadt Noise Dataset (DND) [39]. . . .	15
2.5	Sample image pair patches from Smartphone Image Denoising Dataset [1] (SIDD).	16
2.6	A block diagram of the ground-truth estimation procedure used in SIDD [1].	17
3.1	An overview of the training procedure for the proposed Noise2NoiseFlow framework. Given a pair of independent noisy samples of the same underlying signal, it runs both noisy samples through a denoiser network \mathbf{D} , which outputs the estimated clean signal. We then use the estimated clean image from the first image in place of the true clean signal for the second noisy observation and vice versa. This prevents the denoiser from collapsing into a degenerate solution of an identity transformation. Note that in the paper we formulate the noise model as a distribution over the noisy image \mathbf{f} for ease of notation, though, as shown here, it is common for noise models to be expressed as a distribution of the residual noise $\mathbf{N} = \mathbf{f} - \mathbf{I}$. These two formulations are equivalent.	22
3.2	Overview of the proposed R2RFlow formulation. A single noisy observation is <i>re-corrupted</i> according to equation 3.7 resulting in $\mathbf{f}_{\text{input}}$ and $\mathbf{f}_{\text{target}}$. The input recorruption $\mathbf{f}_{\text{input}}$ is denoised through the denoiser \mathbf{D} . The denoiser loss is calculated as the mean squared error between the denoised image $\mathbf{D}(\mathbf{f}_{\text{input}})$ and $\mathbf{f}_{\text{target}}$ (Eq. 3.9). The negative log-probability of the residual noise $\mathbf{f}_{\text{input}} - \mathbf{D}(\mathbf{f}_{\text{input}})$ is then calculated through the noise model, conditioned on the estimated clean image $\mathbf{D}(\mathbf{f}_{\text{input}})$ (Eq. 3.8).	26

4.1	Noise synthesis samples from (a) the AWGN model, (b) Camera NLF, (c) Calibrated P-G [54], (d) Noise Flow [2], and our proposed method, Noise2NoiseFlow, compared to the (f) real noise in SIDD. Our samples are closer to real noise in terms of both visual perception and the KL divergence metric. Codes on the left indicate [camera]-[ISO]-[brightness]. For Calibrated P-G [54] calibrated parameters for SIDD only include three camera sensors (IP, GP, and S6) so some synthesis results are not available.	34
4.2	Denoising results from (b) DnCNN-supervised, (c) Noise2Noise trained with DnCNN, and (d) Noise2NoiseFlow on samples from SIDD-Medium testing data. The codes on the left indicate the ISO level as well as the lighting condition.	37
4.3	Negative log-likelihood per dimension and PSNR results on test data as a function of the regularization term λ . Cross-sample loss refers to training with Eq. 3.3 and self-sample loss refers to training with Eq. 4.1. Cross marks indicate the loss at the last epoch in cases where training failed as evidenced by spikes in NLL and significant drops in PSNR. All experiments with the self-sample loss and 2^{16} ultimately diverge.	39
4.4	Convergence curve of the two parameters (θ_1 and θ_2) of the NLF model for a specific camera sensor and ISO level. <i>NF Parameter</i> corresponds to the parameters learned by a supervised Noise Flow model and <i>Reconstruction</i> Corresponds to the NLF parameters learned by a Noise2NoiseFlow model from synthetic data generated by the supervised Noise Flow model. As evidenced by the figures, the model can successfully retrieve the parameters.	42
4.5	Noise synthesis samples from (a) the AWGN model, (b) Camera NLF, (c) Calibrated P-G [54], (d) Noise Flow [2], and our proposed method, Noise2NoiseFlow, compared to the (f) real noise in SIDD for patches where Noise2NoiseFlow has the worst D_{KL} numbers.	44

Nomenclature

AWGN Additive White Gaussian Noise

CMOS Complementary Metal-Oxide Semiconductor

DND Darmstadt Noise Dataset

DNN Deep Neural Network

GAN Generative Adversarial Networks

HGN Heteroscedastic Gaussian Noise

IID Independent and Identically Distributed

ISO International Organization for Standardization

KLD Kullback–Leibler Divergence

MSE Mean squared error

NLF Noise-Level Function

NLL Negative Log-Likelihood

P-G Poisson-Gaussian

PSNR Peak Signal-to-Noise Ratio

SDN Signal-Dependent Noise

SIDD Smartphone Image Denoising Dataset

sRGB Standard Red Green Blue

SSIM Structural Similarity Index Measure

SURE Stein's Unbiased Risk Estimate

Chapter 1

Introduction

Image noise modeling is a long-standing problem in computer vision that has relevance for many applications [10, 11, 30, 29, 6, 40]. Recently, data-driven noise models based on deep learning have been proposed [15, 2, 31]. Unfortunately, these models generally require clean (*i.e.*, noise-free) images, which are practically challenging to collect in real scenarios [1]. In this work, we propose a new approach, Noise2NoiseFlow, which can accurately learn noise models without the need for clean images. Instead, only pairs of noisy images of a fixed scene are required.

While efforts are made to reduce noise during capture, post-capture modeling is a critical piece of many downstream tasks and in many domains, large amounts of noise are intrinsic to the problem—for example, astrophotography and medical imaging. As a result, noise is an integral and significant part of signal capture in many imaging domains, and modeling it accurately is critical. For instance, noise model estimation is necessary for removing fixed pattern effects from CMOS sensors [14] and enhancing video in extreme

low-light conditions [47]. Noise models can also be used to train downstream tasks to be robust in the presence of realistic input noise. It can also be used as an augmentation method even in more classical learning tasks, such as object recognition, or other topics such as contrastive learning. Most naturally, they can also be used to train noise reduction algorithms without the need to collect pairs of clean and noisy images [2, 36, 54]. A generative noise model is also beneficial in the way that it acts as a regularizer to prevent the model from overfitting, compared to training on a static real-noise dataset. An example of such an experiment is mentioned in [2] where training a denoiser on synthetic noise generated from Noise Flow outperforms a denoiser trained on real noise image data. However, as mentioned in [55, 44, 3] denoisers trained with unrealistic noise models—for example, simple Gaussian noise—may not perform well on real data.

Early attempts at noise modeling were limited and failed to fully capture the characteristics of real noise. Simple IID Gaussian noise (also called a homoscedastic Gaussian noise) ignores the fact that photon noise is signal-dependent. Heteroscedastic Gaussian noise (*e.g.*, [10]) captures this by modeling noise variance as a linear function of clean image intensity but does not take into account the spatial non-uniformity of noise power, amplification noise, quantization effects, and more. More recently, Noise Flow [2] was proposed as a new parametric structure that uses conditional normalizing flows to model noise in the camera imaging pipeline. This model is a combination of unconditional and conditional transformations that map simple Gaussian noise into a more complex, signal-, camera-, and ISO-dependent noise distribution and outperformed previous baselines by a large margin in the framework of the normalizing flows [24].

Unfortunately, the methods discussed above require supervised noise data—namely,

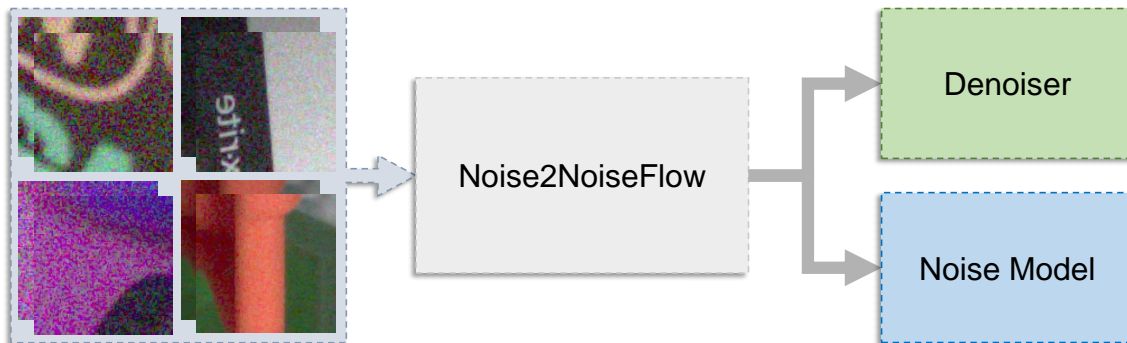


Figure 1.1: Overview of Noise2NoiseFlow. Given pairs of noisy images of the same scene, Noise2NoiseFlow simultaneously trains both a noise model and a denoiser.

pairs of clean and noisy images—in order to learn the noise model. Gathering supervised data consisting of corresponding clean and noisy images can be challenging [1, 39, 3, 49] and is a limiting factor in the realistic characterization of noise. It can be more challenging in tasks such as medical imaging, or environments with moving objects, or non-constant lighting and other conditions. This is even worse for other downstream tasks, which typically require large amounts of data for training.

In the context of image denoising specifically, there has been significant recent interest in methods that avoid the need for supervised data, either from careful collection or synthesis. The well-known BM3D method [8] proposed a denoising scheme based on transform domain representation without clean image correspondence. However, the similar patch search step makes the inference time complexity inefficient for large-scale datasets. Recently, Lehtinen et al. [28] introduced the Noise2Noise framework, which allowed for the training of a denoiser given pairs of noisy images of the same underlying image signal. Following this work, several others were proposed aiming to further reduce the data requirements; in particular, Noise2Void [25] and Noise2Self [4] allow training of a denoiser

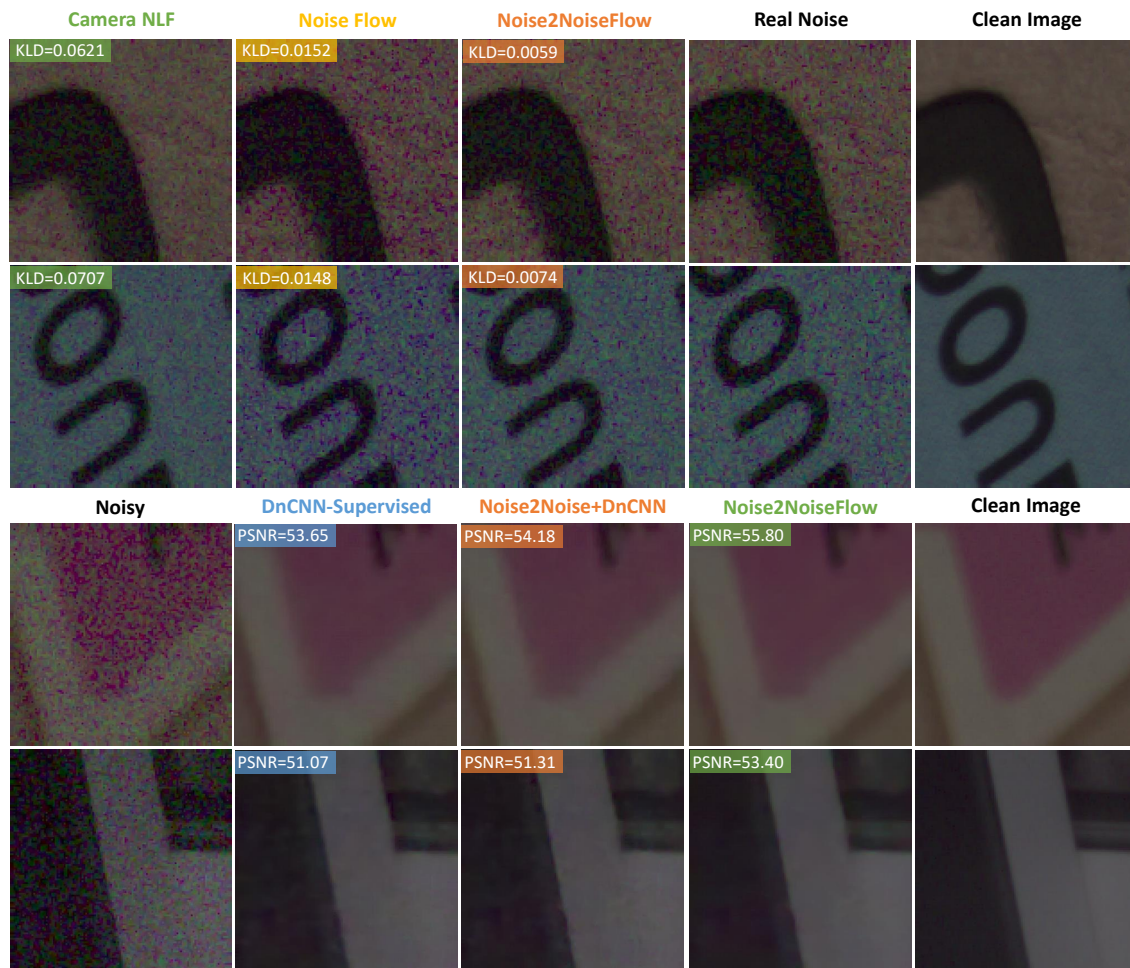


Figure 1.2: Sample results from our Noise2NoiseFlow model. (top) Noise synthesis results from Camera NLF, Noise Flow, and Noise2NoiseFlow compared to the real noise in the SIDD dataset. Noise generated by Noise2NoiseFlow is the most similar to the real noise both visually and in KL divergence but without requiring clean, noise-free images. (bottom) Example denoising results from the jointly trained denoiser compared to its supervised DnCNN baseline, and a DnCNN trained with Noise2Noise loss.

with only individual noisy images by forcing the denoiser to predict the intensity of each pixel using only its neighbours. Other methods attempted to add additional noise to noisy input images [37, 35, 50] or use unpaired images in a GAN framework [5, 7, 16, 19, 20], where the GANs are typically trained to synthesize noise for a downstream denoiser training. However, in all cases, these methods are aimed primarily at denoising instead of noise modeling.

In this work, we aim to leverage these recent advances in training denoisers without direct supervision in the context of noise modeling. Specifically, we extend the Noise2Noise framework to train a noise model with pairs of independently sampled noisy images rather than clean data. The resulting approach, called Noise2NoiseFlow, produces both a denoiser and an explicit noise model, both of which are competitive with or out-perform fully supervised training of either model individually. Figure 1.1 illustrates the overall approach and Figure 1.2 show sample outputs from our model.

1.1 Contributions

Here we list our contributions in this work.

- We introduce a generic framework for jointly training a noise model and a denoiser without clean images, which is not limited to a specific architecture for the noise model or the denoiser. We show that it is insensitive choices of noise models and denoisers until the noise model has a tractable density function and the whole function is differentiable.
- We introduce Noise2NoiseFlow, which is a special case of our generic framework

where a DnCNN [51] is used as the denoising DNN architecture, and Noise Flow as the noise model. We make sure that the noise modeling performance and the sample quality are on par with its supervised baseline and outperform other baselines.

- Although the ultimate goal of the model is noise modeling, we evaluate the denoising performance of our jointly-trained denoiser. We observe a significant improvement in the denoising performance compared to a Noise2Noise denoiser indicating that the joint training framework has a positive impact on the denoising objective.
- We also explore a possible solution for noise modeling with single noisy images by formulating a joint training using the R2R [37] method. We dub this model as R2RFlow and evaluate it in noise density estimation, sample quality, and denoising performance, which show a similar trend as the denoising domain. While Noise2NoiseFlow performs on par with its supervised baseline, R2RFlow lags behind in both density estimation and sample quality.

Chapter 2

Background and Related Work

In this chapter, we review existing methods in noise models, attempts at noise reduction, and noise dataset gathering. We start from simple parametric models and then review recent more sophisticated models based on deep learning architectures. Further, we investigate recent advances in supervised image denoising with deep models and pinpoint their overall training procedure. Then, we identify gathering noisy-clean image pairs as a requirement for training such models and shed light on the merits of training with unrealistic noise samples. Additionally, we investigate recent datasets in the image noise literature, and the procedure needed for gathering them. Finally, we review the recent advances in methods that obviate the need for clean images in training a denoising deep neural network.

2.1 Image Noise Modeling

Image noise can be described as an undesirable corruption added to an underlying clean signal. Formally,

$$\mathbf{f} = \mathbf{I} + \mathbf{N}; \quad (2.1)$$

where \mathbf{I} is the underlying and mostly unobserved clean image and \mathbf{N} is the unwanted noise corrupting the signal, and their addition results in the noisy observation \mathbf{f} . Different noise models are then defined by the choice of distribution assumed for \mathbf{N} . Early attempts at noise modeling were limited to simple models. Additive White Gaussian Noise model (AWGN), also known as the Homoscedastic Gaussian Noise model assumes that noise $\mathbf{N}(x; y)$ at each pixel location $\mathbf{I}(x; y)$ is drawn from a zero-mean Gaussian distribution with some fixed variance, for all pixel locations. Namely

$$\mathbf{N}(x; y) \sim \mathcal{N}(0; \sigma^2); \quad (2.2)$$

While being widely used in the community, even in recent years, this model fails to capture significant aspects of real noise, most prominently the signal-dependent variance, which is a result of the inherent Poisson shot noise [30, 34]. To account for the signal-dependency of the image noise, a Poisson distribution is adapted. Namely, it assumes that the noise at each pixel is drawn from the following distribution

$$\mathbf{N}(x; y) \sim \mathcal{P}(\mathbf{I}(x; y)) \quad \mathbf{I}(x; y); \quad (2.3)$$

Neither the Homoscedastic Gaussian nor the Poisson model can capture the image noise sources perfectly, as the image noise is a combination of both signal-independent and signal-dependent factors. Therefore, a Poisson-Gaussian noise model is usually adopted [10, 11, 32, 54]. Namely, the noise at each pixel is drawn according to the following distribution

$$\mathbf{N}(x; y) = P(\mathbf{I}(x; y)) = \mathbf{I}(x; y) + N(0; \sigma^2); \quad (2.4)$$

A more widely used alternative is when the Poisson is replaced by a Gaussian distribution, named as Noise-Level Function (NLF) also known as the Heteroscedastic Gaussian Noise model. This model assumes that the noise at each pixel is drawn as

$$\mathbf{N}(x; y) = N(0; \sigma^2(\mathbf{I}(x; y))) \quad (2.5)$$

with

$$\sigma^2(\mathbf{I}(x; y)) = \sigma_1 \mathbf{I}(x; y) + \sigma_2 \quad (2.6)$$

where σ_1, σ_2 are parameters. Recent work has shown that NLF parameters from camera manufacturers are often poorly calibrated [54]. The signal-dependency of the NLF and Poisson-Gaussian model captures various image noise sources such as photon noise. However, they neglect important noise characteristics that may exist in an imaging pipeline, including spatial correlation, defective pixels, fixed pattern noise, clipping, quantization, and more.

More recently, researchers have turned towards introducing deep neural network architectures to address the above issue. DNN-based models use an implicit parameterization to account for the missing noise sources in conventional explicit models. Noise Flow [2]

is one example of such a model that combines the power of Normalizing Flows [42, 9, 23] with the domain knowledge to design a customized architecture that achieves state-of-the-art performance in noise modeling and sample quality compared to previous baselines. DeFlow [48] is another example of such a model that handles a broader range of image degradations beyond traditional noise. Other approaches consider mixture models or Generative Adversarial Networks (GAN) [12] to simulate noisy and clean images in the context of denoiser training [5, 7, 56, 16, 19, 20, 15]. However, these models are typically focused on denoising as opposed to noise modeling. Further, GANs do not have tractable likelihoods, making the quality of the synthesized noise difficult to assess. Most importantly, the above methods require clean images, and potentially pairs of noisy and corresponding clean images for training. In this work, we construct a formulation that explicitly trains a noise model without the need for clean images. Because of the flexibility and generality of the normalizing flow framework and the quality of its results, we will focus on the Noise Flow model [2] here, though, as we will discuss, other choices are possible.

Here we will discuss technical details of Noise Flow in more depth. Noise Flow uses conditional normalizing flows transformation to capture the complex camera image noise characteristics conditioned on the clean image, ISO level, and camera sensor type. Normalizing flows are a class of generative models that transform an unknown complex probability density function in a data domain into a simpler and known distribution, such as an isotropic Gaussian using a series of bijective transformations. They offer an exact evaluation of the probability density function and ensure almost-perfect reconstruction through a series of invertible transformations with known inverse function and Jacobian matrix. Generating a new sample from such a model is as easy as sampling a point from the base

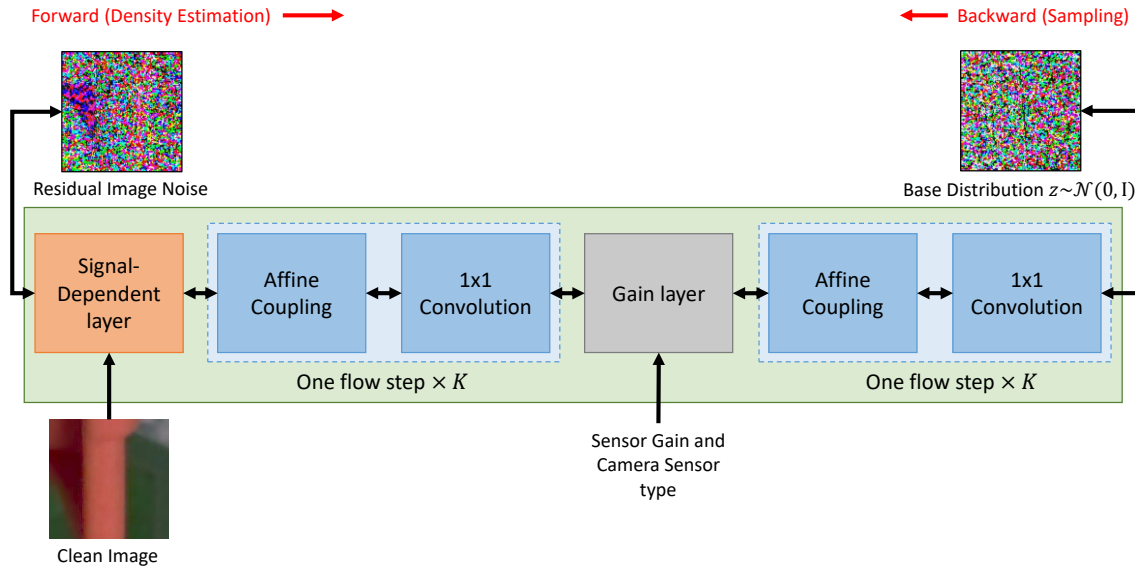


Figure 2.1: An overview of the Noise Flow model proposed by Abdelhamed et al. [2]. The forward pass takes the residual image noise and transforms it into a known base distribution, which in this case is an isotropic Gaussian model. The transformation is conditioned on the clean image, sensor type, and gain factor (ISO level). The backward pass takes a sample from the isotropic Gaussian distribution and transforms it into a sample from the image noise distribution, conditioned on the clean image, sensor type, and sensor gain.

distribution, and transforming using the inverse pass. In the case of Noise Flow, the data domain is defined as the camera image noise, which is transformed into a simple isotropic Gaussian latent space using specialized conditional transformations combined with well-known unconditional flow layers. Noise Flow can be seen as a strict generalization of HGN due to its use of a signal-dependent transformation layer, which we will discuss. However, unlike HGN, Noise Flow is capable of capturing non-Gaussian distributions and complex spatial correlations.

Noise Flow uses two conditional transformations, namely Gain and Signal-Dependent

Noise (SDN) layers. The forward direction of the Gain layer is computed as

$$f(\mathbf{x}) = (\text{ISO}) \cdot \mathbf{x}; \quad (\text{ISO}) = u(\text{ISO}) \cdot \text{ISO}; \quad (2.7)$$

where the function u compensates for the strict scaling in the ISO values, and \cdot is the point-wise multiplication. The intuition comes from the domain knowledge that the gain factor (or roughly the ISO level) amplifies both the signal and the noise. The SDN layer is designed to be a generalization of the HGN. In theory, it is estimated by a Poisson-Gaussian model, and in practice, it is mostly substituted with a zero-mean Gaussian model with signal-dependent variance, as discussed before (Eq. 2.5). The SDN forward pass is computed as

$$f(\mathbf{x}) = \mathbf{s} \cdot \mathbf{x}; \quad \mathbf{s} = \frac{\rho}{(\sigma_1 \mathbf{I} + \sigma_2)}; \quad (2.8)$$

where \mathbf{I} is the clean image, and σ_1 and σ_2 are learnable parameters formulated as $\sigma_1 = \exp b_1$ and $\sigma_2 = \exp b_2$ to ensure the overall expression is positive. The parameters b_1 and b_2 are initialized such that the overall forward step is an identity transformation.

Noise Flow combines the two conditional layers discussed above with an unconditional block composed of affine coupling transformations [9] and invertible 1×1 convolutions [22] to capture complex spatial correlations and other sources of the noise. One flow step in the unconditional block corresponds to an affine coupling transformation followed by an invertible 1×1 convolution. Figure 2.1 shows the overall architecture of the Noise Flow model.

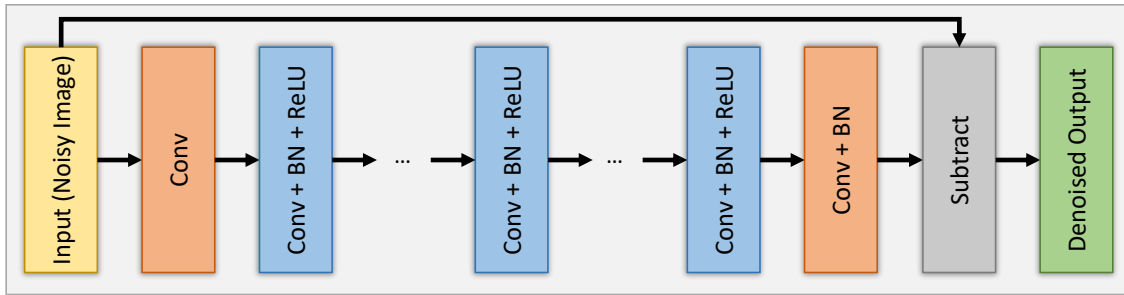


Figure 2.2: The overall architecture of the DnCNN [51] model. “BN” denotes Batch Normalization [18].

2.2 Supervised Image Denoising

Image noise reduction has been a long-standing topic of study in computer vision [26, 29, 53, 6, 40, 8]. Image denoising on supervised data is the task of learning to predict the noise-free image given a noisy observation, on a set of noisy-clean image pairs. Here we focus on recent methods that have found success by leveraging large training sets and deep learning architectures [51, 43]. DnCNN [51] and U-Net [43] are two recent examples that have found success in denoising performance and are widely used for benchmarking deep image denoisers. DnCNN uses a residual convolutional DNN for denoising while U-Net is designed as a downsampling network followed by an upsampling. While different modifications have been made to these two architectures for various purposes, Figures 2.2 and 2.3 show an overview of their base models, respectively. FFDNet [52] and CBDNet [13] are two other examples of recent architectures introduced to benchmark image denoising methods.

Training DNN-based supervised denoisers are characterized by regressing, typically with a convolutional neural network, from a noisy image observation to its clean counter-

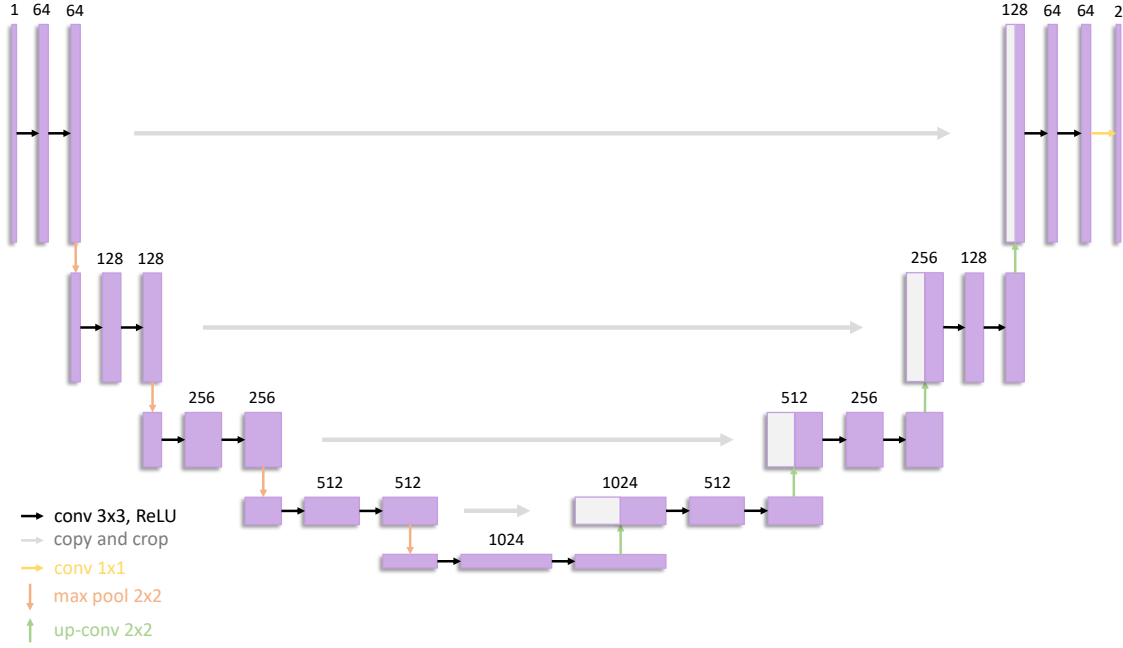


Figure 2.3: Overview of the U-Net architecture introduced by Ronneberger et al. [43]. The figure is re-produced from the original paper. Note that the architecture they introduce was originally used for biomedical image segmentation, and not image denoising.

part. Given a training set $D = \{(\mathbf{t}^{(i)}; \mathbf{I}^{(i)})\}_{i=1}^N$ of noisy images \mathbf{t} and their corresponding clean images \mathbf{I} , learning of a denoiser \mathbf{D} is then formulated as minimizing

$$\sum_{i=1}^N L(\mathbf{D}(\mathbf{t}^{(i)}; \cdot); \mathbf{I}^{(i)}); \quad (2.9)$$

where L is typically an L_1 or L_2 norm and \mathbf{D} is a deep neural network with parameters θ . Deep neural networks require massive amounts of data to train. These models assume a noise dataset of noisy-clean image pairs are provided for training.



Figure 2.4: A sample image pair from the Darmstadt Noise Dataset (DND) [39].

2.3 Noise Datasets

The noise models, as well as the supervised denoising methods discussed above, require access to clean images to estimate the parameters, especially, deep neural networks with a large number of learnable parameters. Image denoising methods used to benchmark their models on unrealistic synthetic image noise, *e.g.*, a simple Gaussian noise. However, simple synthetic noise may not be a good representation of real noise distribution, and it would be easy for the denoisers to overfit [13]. Multiple researchers have reported such a phenomenon [55, 44, 3]. According to their observation, denoisers trained on unrealistic synthetic noise may not perform well on denoising real noisy images. This necessitates having a dataset of image noise from realistic scenes.

Image noise is an unwanted corruption added to the image at the time of measurement. It is also an inevitable part of the imaging system. The amount of noise may be controlled

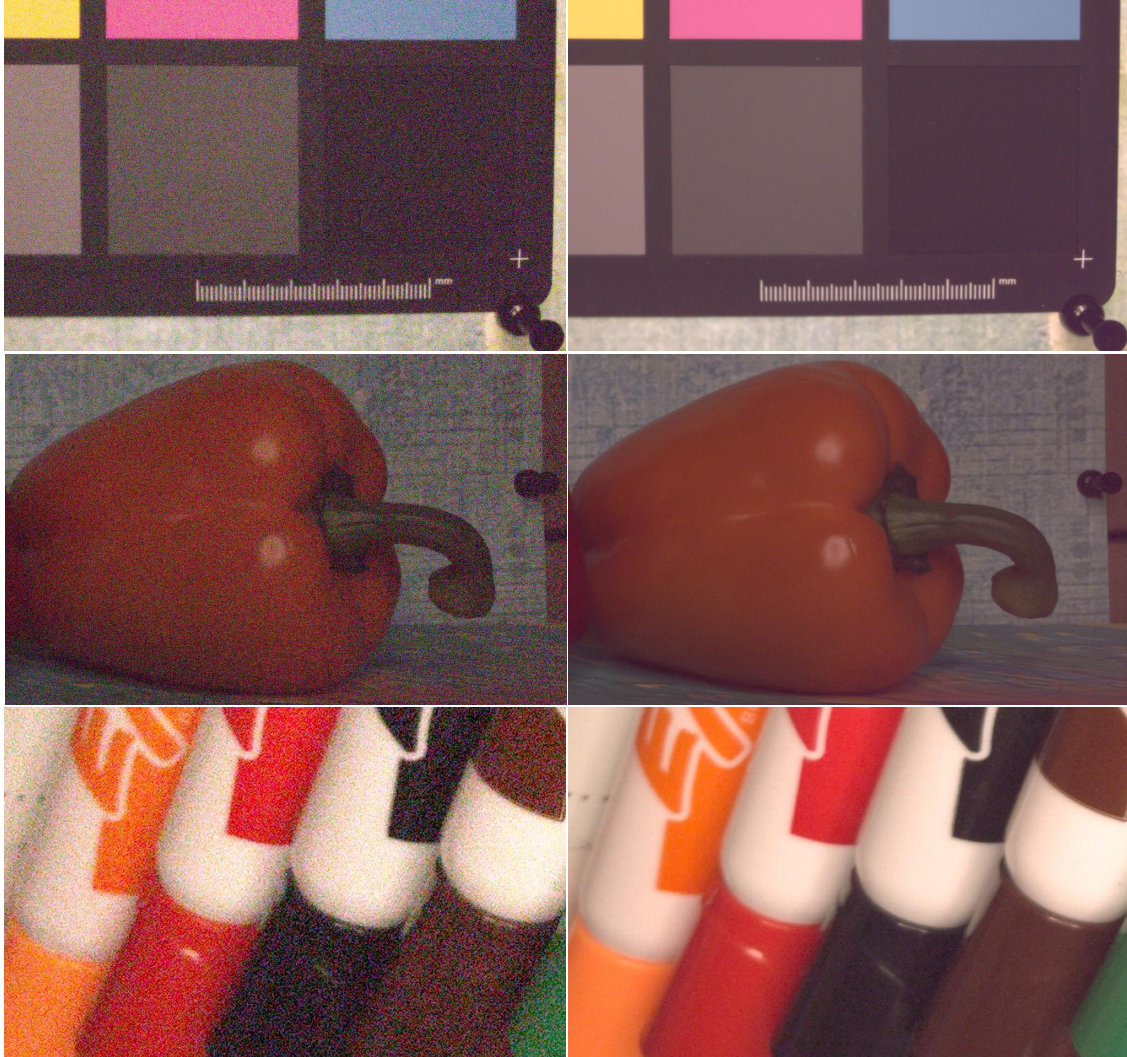


Figure 2.5: Sample image pair patches from Smartphone Image Denoising Dataset [1] (SIDD).

through different setups, however, capturing a noise-free image is impossible. Therefore, we need to estimate the clean image corresponding to a noisy image observation. The estimation of clean images is a challenging problem [1, 39, 3].

It requires a long process of capturing with careful consideration. The Smartphone

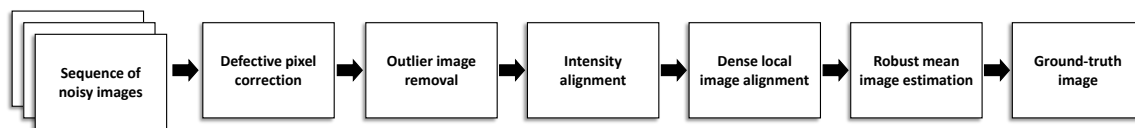


Figure 2.6: A block diagram of the ground-truth estimation procedure used in SIDD [1].

Image Denoising Dataset [1] (SIDD) for example, captures 150 images for each scene resulting in 150 noisy observations. A heavy aligning process is then applied to the 150 captured images to correct for the pixel shifts. The final estimate of the clean image is obtained by averaging the aligned noisy observation coming from the assumption that most of the noise components are zero-mean. Figure 2.6 shows an overview of the procedure used in SIDD for ground-truth estimation. The authors of the Darmstadt Noise Dataset [39] (DND) take a different approach of short and long exposures to create noisy-clean image pairs. Figures 2.5 and 2.4 show sample image pairs from SIDD and DND, respectively.

Both of the procedures described above are challenging and tedious. Also, they are only applied to very controlled environments, as lighting and other capturing conditions need to be constant between different captures. It is also more challenging, and even impossible in the presence of dynamic objects. This would limit the practicality of existing models as they rely on clean images.

2.4 Image Denoising without Clean Images

As discussed above, training denoisers on synthetic noise samples from simple models have shown not to be effective [49]. Furthermore, gathering a real-world image noise dataset is also challenging. Motivated by this constraint, several notable approaches have

recently been explored to remove this requirement. Most relevant to this work is the Noise2Noise framework, proposed by Lehtinen et al. [28]. Rather than requiring clean/noisy pairs of images, it simply requires two noisy observations of the same underlying clean signal. This idea also comes from the assumption that the image noise is roughly zero-mean. Therefore, regressing from a noisy image to another noisy observation of the same scene (rather than the clean image) is similar to the generalization techniques used in other tasks in the literature. The authors prove that in expectation, the distribution of the gradients computed for their objective matches that of the supervised training. Given a dataset of noisy image pairs $f(\mathbf{t}_1^{(i)}; \mathbf{t}_2^{(i)})_{i=1}^N$, the Noise2Noise framework optimizes the loss function

$$\prod_{i=1}^N L(\mathbf{D}(\mathbf{t}_1^{(i)}; \cdot); \mathbf{t}_2^{(i)}) + L(\mathbf{D}(\mathbf{t}_2^{(i)}; \cdot); \mathbf{t}_1^{(i)}) : \quad (2.10)$$

That is, the second noisy image is used as the target for the denoiser of the first and vice versa. Perhaps surprisingly, training with this objective is still able to produce high-quality denoising results, despite the lack of access to clean images [28]. The results show that the denoising performance is highly on par with the supervised baseline. In this work, we aim to explore the generalization of this approach to noise modeling.

To further relax the requirements, other methods have also been proposed that take different approaches to train denoisers with single noisy image observations. For instance, Noise2Void [25], and Noise2Self [4] and the work from Laine et al. [27] lie under the *data augmentation* methods that adopt a blind-spot strategy to avoid degeneracy to a trivial identity solution. They ask the model to predict a certain pixel using its surrounding pixels. The blind-spot strategy can also be interpreted as a re-corruption of the target image

with multiplicative Bernoulli noise [37]. Other models such as Noisier2Noise [35], and Noisy-as-Clean [50] augment the data by adding additional noise to the input image. More recently, Pang et al. [37] has adopted a similar strategy, namely Recorruped2Recorruped (R2R) of augmenting both the input and output by adding noise so that the overall objective in theory matches that of supervised learning.

Other methods regularize the predictions of a denoising DNN. In Self2Self [41] as an example, authors have proposed a dropout-based framework in both training and testing to control the bias and variance of the model predictions. An early stopping has been adapted in Deep Image Prior [46] to avoid overfitting. Two SURE-based methods [33, 45] penalize divergence in the model predictions. Another approach to learning unsupervised denoisers is to train GANs to synthesize noise in order to train a subsequent denoiser [5, 7]. In particular, Cha et al. [5] propose a three-step training procedure of training a GAN to generate synthetic image noise to further augment the single image data in order for training a subsequent denoising DNN.

The unsupervised denoising models described above have all been evaluated against their supervised baselines. Noise2Noise is special in that it avoids the clean image requirement with minimal drops in performance compared to its supervised baseline. It even can surpass the supervised performance in certain cases, as we will show in this work. However, the unsupervised methods, despite removing the need for any source of supervision, still lag behind the supervised baselines and have more room for improvement. Perhaps the loss of such important information limits the amount by which they can get close to the supervised models.

While efforts have been made to relax the clean image requirement in the denoising

domain, this has been relatively unexplored in noise modeling. The existing methods typically rely on noisy-clean paired data, which limits their practical use. In this work, we explore the possibility of applying a similar strategy in noise modeling to make it more practical. We show that with a similar idea, we can make noise modeling possible with only noisy-noisy image pairs, similar to that of Noise2Noise. Then, we show that we can even go a step further and train a noise model with no supervision (single noisy images), but need to bear a significant performance drop, a result similar to that of denoising.

Chapter 3

Noise2NoiseFlow

In this chapter, we define our approach to learning a noise model with weak supervision—namely, through the use of only pairs of noisy images. Our model has two main components, a denoiser \mathbf{D} , and a conditional noise model $p_{\mathbf{t}}$. The denoiser $\mathbf{D}(\cdot; \cdot)$ learns to predict the clean image \mathbf{I} given a noisy image, \mathbf{t} , as input. The noise model learns the model of a noisy image given the clean image \mathbf{I} , $p_{\mathbf{t}}(\cdot|\mathbf{I}; \cdot)$. The denoiser and noise model have parameters θ and ϕ respectively. Our goal is to learn the distribution $p_{\mathbf{t}}(\mathbf{t}|\mathbf{I})$ —namely, the distribution of noisy image conditioned on the clean image—without explicitly requiring \mathbf{I} .¹ To do this, we propose to use the output of the denoiser as an estimate of the clean image—That is,

$$\mathbf{I} \approx \hat{\mathbf{I}} = \mathbf{D}(\mathbf{t}; \theta): \tag{3.1}$$

¹Note that this is equivalent to learning the distribution of the noise conditioned on the clean image by simply shifting the distribution of noise by the clean image. As the conditional flow model has access to clean images, it can easily subtract the clean image from the noisy input. Therefore, learning the noisy image conditioned on the ground-truth clean image is equivalent to learning the conditional residual noise distribution. We opt for this only for ease of notation.

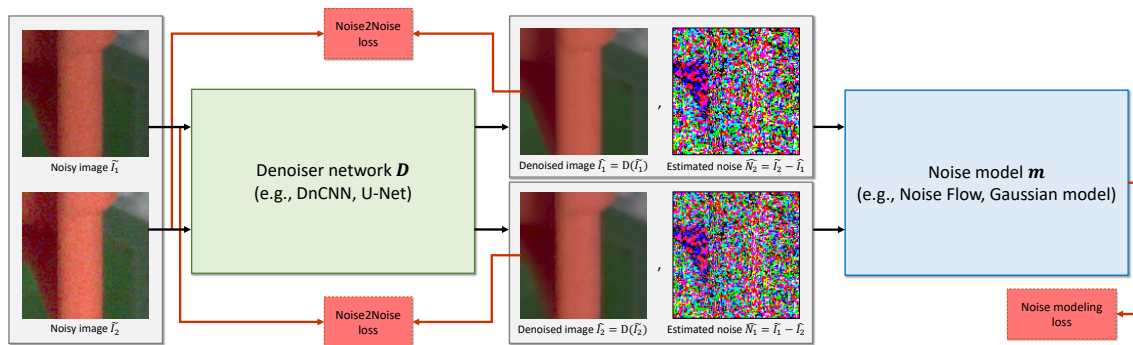


Figure 3.1: An overview of the training procedure for the proposed Noise2NoiseFlow framework. Given a pair of independent noisy samples of the same underlying signal, it runs both noisy samples through a denoiser network D , which outputs the estimated clean signal. We then use the estimated clean image from the first image in place of the true clean signal for the second noisy observation and vice versa. This prevents the denoiser from collapsing into a degenerate solution of an identity transformation. Note that in the paper we formulate the noise model as a distribution over the noisy image \mathbf{t} for ease of notation, though, as shown here, it is common for noise models to be expressed as a distribution of the residual noise $\mathbf{N} = \mathbf{t} - \mathbf{I}$. These two formulations are equivalent.

We could in principle then learn $p_{\mathbf{t}}$ by minimizing $-\log p_{\mathbf{t}}(\mathbf{t}; \hat{\mathbf{I}})$ with respect to the noise model parameters θ . However, this requires a well-trained denoiser, which, in turn, typically requires access to clean images to train. Further, if we tried to simultaneously train the denoiser and noise model, there is a trivial singular optimum where the denoiser converges to the identity and the noise model converges to a Dirac delta at zero.

Drawing inspiration from the Noise2Noise framework [28], we instead assume we have access to pairs of noisy observations $\mathbf{t}_1; \mathbf{t}_2$ which both have the same underlying clean signal, \mathbf{I} . That is,

$$\mathbf{t}_1 = \mathbf{I} + \mathbf{N}_1; \quad \text{and} \quad \mathbf{t}_2 = \mathbf{I} + \mathbf{N}_2; \quad (3.2)$$

where \mathbf{N}_1 and \mathbf{N}_2 are independent samples of noise. Then, given the pairs of noisy images, we can use the denoiser applied to one image to estimate the clean image for the other image in the pair. That is, we propose to optimize the loss

$$L_{nm}(\mathbf{t}_1; \mathbf{t}_2) = -\log p_{\mathbf{t}}(\mathbf{t}_1 | \mathbf{D}(\mathbf{t}_2; \cdot); \cdot) - \log p_{\mathbf{t}}(\mathbf{t}_2 | \mathbf{D}(\mathbf{t}_1; \cdot); \cdot); \quad (3.3)$$

for both the noise model parameters \cdot and the denoiser parameters \cdot . Because the two images are of the same underlying scene, the output of the denoiser should ideally be the same for both noisy images. However, because the two images have independent samples of noise, the denoiser cannot simply collapse to the identity. This is analogous to the Noise2Noise objective, where the output of the denoiser on one image is used as the target for the other image in the pair. Another choice would be to use each denoised image as the estimated clean image for its corresponding noisy image. That is, a more natural alternative would be to optimize

$$L'_{nm}(\mathbf{t}_1; \mathbf{t}_2) = -\log p_{\mathbf{t}}(\mathbf{t}_1 | \mathbf{D}(\mathbf{t}_1; \cdot); \cdot) - \log p_{\mathbf{t}}(\mathbf{t}_2 | \mathbf{D}(\mathbf{t}_2; \cdot); \cdot); \quad (3.4)$$

rather than the formulation proposed in Equation 3.3. As mentioned above, it is assumed that both noisy observations share a similar latent clean image, and are corrupted with two i.i.d. noise samples (Eq. 3.2), drawn from the distribution that our model aims to estimate. Therefore, it is natural to expect that in theory, the output of the denoiser for both noisy images are identical—namely, $\mathbf{D}(\mathbf{t}_1; \cdot) = \mathbf{D}(\mathbf{t}_2; \cdot)$, although we do not impose such constraint in our loss formulation. As a result, it is possible to formulate according to the Equation 3.3. We used the formulation in Equation 3.3 rather than the other to avoid

both our denoiser and noise model to collapse into an identity transformation. With the denoiser collapsed to identity, the estimated residual noise would be almost zero, which is very easy for the noise model to capture as a zero vector has the highest probability in the base distribution of Noise Flow, which is an isotropic Gaussian.

In practice, we find it beneficial to include the Noise2Noise objective function to stabilize the training of the denoiser together with the noise model objective. That is, we propose to train the denoiser and noise model jointly with the loss

$$L(\mathbf{t}_1; \mathbf{t}_2) = L_{nm}(\mathbf{t}_1; \mathbf{t}_2) + L_{dn}(\mathbf{t}_1; \mathbf{t}_2); \quad (3.5)$$

where

$$L_{dn}(\mathbf{t}_1; \mathbf{t}_2) = k\mathbf{D}(\mathbf{t}_1; \mathbf{t}_2) + k\mathbf{D}(\mathbf{t}_2; \mathbf{t}_1); \quad (3.6)$$

is the Noise2Noise loss. We investigate the benefits of our choices in our ablation studies in the next chapter. Given a dataset of pairs of noisy images, $D = \{(\mathbf{t}_1^{(i)}, \mathbf{t}_2^{(i)})\}_{i=1}^N$, we optimize the loss over the set of pairs

$$\sum_{i=1}^N L(\mathbf{t}_1^{(i)}, \mathbf{t}_2^{(i)});$$

where the optimization can be done with a stochastic optimizer. In this work we use Adam [21].

Figure 3.1 shows an overview of the proposed approach. We note that the formulation is generic to the choice of denoiser and noise model, requiring only that the noise model’s density function can be evaluated and that both the noise model and denoiser can

be differentiated as needed. In the experiments that follow we primarily use the DnCNN architecture [51] for the denoiser, as it is a standard denoiser architecture based on residual connections and convolutional layers. For the noise model, we primarily focus on Noise Flow [2] due to its flexibility and tractability and, consequently, dub our proposed method Noise2NoiseFlow. However, we also explore other choices for these components, such as a U-Net architecture for the denoiser and the heteroscedastic Gaussian noise model.

3.1 R2RFlow

Here we have proposed a novel approach to noise model training by coupling the training of a noise model with a denoiser and based on the Noise2Noise framework. This naturally raises the question of whether a noise model could be trained with only individual noisy images, particularly given the success of such approaches for denoisers. All of these approaches aim to prevent the denoiser from collapsing into the degenerate solution of an identity transformation, by either using a blind-spot network architecture (e.g, Noise2Void [25] and Noise2Self [4]), or adding additional noise to the input images (e.g, Noisier2Noise [35], Noisy-as-Clean [50], and R2R [37]). We will investigate such behaviour in the experiments chapter on the formulation proposed in equation 3.4. To investigate the idea of training a noise model with single noisy images, we considered using the R2R [37] framework, which, given a single noisy image \mathbf{r} , generates two new noisy images as

$$\mathbf{r}_{\text{input}} = \mathbf{r} + \mathbf{D}^T \mathbf{z}; \quad \mathbf{r}_{\text{target}} = \mathbf{r} - \mathbf{D}^{-1} \mathbf{z}; \quad (3.7)$$

Figure 3.2: Overview of the proposed R2RFlow formulation. A single noisy observation is re-corrupted according to equation 3.7 resulting in $\mathbf{r}_{\text{input}}$ and $\mathbf{r}_{\text{target}}$. The input re-corruption $\mathbf{r}_{\text{input}}$ is denoised through the denoiser. The denoiser loss is calculated as the mean squared error between the denoised image $\mathbf{D}(\mathbf{r}_{\text{input}})$ and $\mathbf{r}_{\text{target}}$ (Eq. 3.9). The negative log-probability of the residual noise $\mathbf{D}(\mathbf{r}_{\text{input}})$ is then calculated through the noise model, conditioned on the estimated clean image $\mathbf{D}(\mathbf{r}_{\text{input}})$ (Eq. 3.8).

where \mathbf{z} is drawn from $\mathcal{N}(0; \mathbf{I})$, and $\mathbf{D} = \mathbf{I} + \sigma^2 \mathbf{A}^{-1}$ is an invertible matrix with scale parameter σ^2 .

We modify our loss functions to utilize these new images so that

$$L_{\text{nm}} = -\log p_{\mathbf{r}}(\mathbf{r}_{\text{target}} | \mathbf{D}(\mathbf{r}_{\text{input}})); \quad (3.8)$$

and

$$L_{\text{dn}} = k \|\mathbf{r}_{\text{target}} - \mathbf{D}(\mathbf{r}_{\text{input}})\|_2^2. \quad (3.9)$$

We train the model by optimizing the loss

$$L = L_{\text{nm}} + L_{\text{dn}}; \quad (3.10)$$

as described above.

The formulation we propose here is a first-cut at noise model training with individual

noisy images. Figure 3.2 illustrates an overview of the proposed derivation. We use a similar setting to our Noise2NoiseFlow framework, except that here we are inspired by the R2R framework, and thus we dub this formulation **R2RFlow**.

Chapter 4

Experiments

Here we evaluate the performance of the proposed Noise2NoiseFlow approach. We start by explaining the dataset, metrics, and training details. Then, we evaluate our model both in terms of noise quality and denoising against its baselines. To validate the choices we made in our framework, we conduct a set of ablations for various aspects of our model. Then, we propose a formulation for training with individual noisy image samples and evaluate it against our base model and its baselines. Furthermore, we verify the ability of our model in retrieving the learned parameters of a supervised Noise Flow model through a synthetic noise experiment. Finally, we showcase the worst noise samples generated by our model to identify where our model fails to generate a realistic sample, according to the KL-Divergence score.

4.1 Dataset

We make use of the Smartphone Image Denoising Dataset (SIDDD) [1] to assess the accuracy of both our learned noise model and the image denoiser. SIDDD contains images of 10 different scenes consisting of a range of objects and lighting conditions captured with five different smartphone cameras at a range of different ISO levels. The camera sensors are “Samsung Galaxy S6 Edge (S6), Apple iPhone 7 (IP), Google Pixel (GP), Motorola Nexus 6 (N6), and LG G4 (G4)”. The ISO levels range from 50 to 10,000, with most of the samples from the set “100, 400, 800, 1600, and 3200”. 150 captures of each scene instance were taken and carefully aligned to produce a corresponding “clean” image for each noisy image. While our proposed method does not require clean images for training, we do make use of them for a quantitative evaluation against a range of baselines, including methods that require clean image supervision.

Here we use two different subsets of SIDDD—namely SIDDD-Full and SIDDD-Medium. While SIDDD provides both sRGB and rawRGB images, here we only consider the rawRGB images. The sRGB images shown in this chapter are rawRGB images rendered in the sRGB space using the raw-to-srgb conversion pipeline made available as a part of the SIDDD. SIDDD-Full provides 150 different noisy captures for each corresponding clean image. In contrast, SIDDD-Medium contains only a single noisy image for each clean image. To extract the noisy/noisy image pairs of the same clean signal from SIDDD-Full that are required by our method for training, we select pairs of noisy images corresponding to the same clean image. To maximize alignment between the selected two images, we select consecutive images from the 150 available for each scene in SIDDD-Full.

We use SIDDD-Medium to evaluate the performance of our method. Specifically, while

we use noisy/noisy pairs of images extracted from SIDD-Full for training as described above, we evaluate the performance of both the denoiser and the noise model using the noisy/clean image pairs in SIDD-Medium. To test Noise2NoiseFlow against our baselines, we use supervised noisy/clean pairs from the SIDD-Medium dataset.

SIDD contains scenes with ISO levels ranging from 50 to 10,000; however, many of those ISO levels have only a small number of images available. To be consistent with other methods that use SIDD for noise modeling—for example, [2]—we remove images with rare ISO levels, keeping only ISO levels 100, 400, 800, 1600, and 3200. After filtering, approximately 500,000 patches of size 32 pixels are extracted. The extracted patches are separated into training and test sets using the same training and testing split of SIDD scenes that were used in [2]. Approximately 70% of the extracted patches were used for training and the remaining were used for testing.

4.2 Metrics

Denoting $(t; l)$ as a noisy/clean image pair, we evaluate the noise modeling using the negative log-likelihood per dimension $-\frac{1}{D} \log p_t(t; l)$, where D is the total number of dimensions (both pixels and channels) in the input. Negative log-likelihood is a standard evaluation metric for generative models and density estimation, but it is known to be less sensitive to distributions that overestimate the variance of a distribution. To account for this we also evaluate the model using the Kullback-Leibler (KL) divergence metric introduced in [2]. Both NLL and KL divergence is reported in nats. Specifically, given a noisy and clean image, we compute a histogram of both real noise and noise generated by a

model by subtracting the clean image and computing the KL divergence between the two histograms. See [2] for more details on this metric. To evaluate the denoiser, we compute peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM).

4.3 Training Details

We trained all models using the Adam optimizer [21] for 2,000 epochs. We used a value of $\beta = 2^{18}$ in all experiments unless otherwise noted. To speed up convergence and avoid early training instabilities we pre-trained the denoiser on the training set for 2,000 epochs using \mathcal{L}_{dn} (Eq. 3.6) alone for all of the experiments. Note that the denoiser pre-training step is only used to boost the subsequent training procedure, and is not a vital part of the training. Later we mention that training without pre-training will produce a similar result.

The denoiser is a 9-layer DnCNN, the same model as used in Noise Flow [2] denoising experiments, and was the same in all experiments except where noted. Noise Flow was re-implemented in PyTorch [38] and carefully tested for consistency against the original implementation. We used the same architecture for Noise Flow as in the original paper. Joint training used a constant learning rate of 10^{-4} for 2,000 epochs though no improvements were generally observed after 600 epochs.

4.4 Results

Next, we discuss the results and evaluations from our models compared to the baselines. We will cover the results from noise modeling and noise reduction. Then, we mention

other possible combinations of architectures through ablation studies to demonstrate the generality of our framework.

4.4.1 Noise Modeling

We first compare our proposed approach quantitatively to traditional noise models which have been calibrated using supervised, clean images. Table 4.1 compares the results of our model against the camera noise level function (Cam-NLF), a simple additive white Gaussian noise model (AWGN), and Noise Flow [2]. Despite only having access to pairs of noisy images, the proposed Noise2NoiseFlow has effectively identical performance to the state-of-the-art Noise Flow model which is trained on clean/noisy image pairs. To demonstrate the benefit of joint training, we trained a Noise2Noise denoiser [28] on noisy/noisy paired data and use this to denoise images to train a subsequent Noise Flow model. We refer to this as “N2N+NF”.

We also compared our results to the recently released “calibrated Poisson-Gaussian” noise model described in [54]. The results for this comparison in terms of KL divergence can be found in the Table 4.2 for the three cameras reported in the paper [54], as the Calibrated P-G model included noise parameters only for three different sensors: iPhone 7, Samsung Galaxy S6 Edge, and Google Pixel. It is clear that while the Calibrated P-G model improves over the in-camera noise level function, it still lags behind both Noise Flow and Noise2NoiseFlow. We again see that the proposed Noise2NoiseFlow outperforms this very recent method.

Figure 4.1 shows qualitative noise samples generated by Noise2NoiseFlow, as well as other baselines compared to the real noise. The samples are generated for different

Model	NLL	D_{KL}
AWGN	-2.874	0.4815
Cam. NLF	-3.282	0.0578
N2N+NF	-3.459	0.0363
Noise Flow	-3.502	0.0267
Noise2NoiseFlow	-3.501	0.0265

Table 4.1: Negative log-likelihood per dimension \bar{d}_{NLL} results on test data for baseline models and our proposed Noise2NoiseFlow model. Noise2NoiseFlow significantly improves over AWGN and Camera NLF and is on par with the Noise Flow model, while requiring no clean images. It also improves over separately training a Noise2Noise denoiser and NoiseFlow (N2N+NF), demonstrating the value of joint training.

	IP	S6	GP	Agg
AWGN	0.4353	0.4863	0.5865	0.4934
Cam. NLF	0.0513	0.1014	0.0212	0.0596
Calibrated P-G	0.0188	0.0981	0.0332	0.0492
Noise Flow	0.0112	0.0469	0.0180	0.0250
Noise2NoiseFlow	0.0125	0.0444	0.0190	0.0249

Table 4.2: Per camera KL divergence performance of our model Noise2NoiseFlow compared to the baselines on three camera sensors for which the Calibrated P-G model is defined as well as the aggregate results on the test data for these three sensors.

camera sensors, ISO levels, and scenes. The suffix N corresponds to normal light and L corresponds to the low-light conditions. As evidenced by these images, the results from Noise2NoiseFlow are both visually and quantitatively better than other baselines, especially in low-light/high-ISO settings, where other baselines underperform.

4.4.2 Noise Reduction

While the primary goal of this work was noise modeling, it also includes a denoiser as a key component. Here we investigate the performance of the denoiser by evaluating its performance in terms of PSNR on the held-out test set. We compared against three scenarios,

Figure 4.1: Noise synthesis samples from (a) the AWGN model, (b) Camera NLF, (c) Calibrated P-G [54], (d) Noise Flow [2], and our proposed method, Noise2NoiseFlow, compared to the (f) real noise in SIDD. Our samples are closer to real noise in terms of both visual perception and the KL divergence metric. Codes on the left indicate [camera]-[ISO]-[brightness]. For Calibrated P-G [54] calibrated parameters for SIDD only include three camera sensors (IP, GP, and S6) so some synthesis results are not available.

which are reported in Table 4.3. In all cases, the same DnCNN architecture is used. First, we trained the same denoiser architecture by increasing the Noise2Noise [28] loss alone. This is shown in Table 4.3 as “Noise2Noise+DnCNN” and shows that, indeed, the joint noise

model training improves the denoising performance by over 1.2dB, a significant margin in PSNR. Second, we trained a supervised DnCNN model using the corresponding clean image patches for the training set; this is indicated in the table as “DnCNN-supervised”. The supervised DnCNN was trained with MSE using the clean/noisy pairs from SIDD-Medium. Both denoiser pretraining and supervised training used an initial learning rate of 10^{-3} , which was decayed to 10^{-4} at epoch 30, and 10^{-5} at epoch 60. We used orthogonal weight initialization [17] for the denoiser architectures and the same initial weights for the noise model as used in the Noise Flow paper. Also note that as mentioned above, the denoiser pre-training step was used only to boost training under different setups, and is not a vital part of the overall training. Training the original Noise2NoiseFlow model from scratch will also produce almost the same results (L1 : 3:498, D_{KL} : 0:0275 PSNR: 52:65).

Noise2NoiseFlow outperforms the supervised DnCNN by nearly 1.5dB, despite not having access to clean images. Both Noise2Noise+DnCNN and Noise2NoiseFlow outperform this clean-image supervised baseline, suggesting that the increased variety of data available with noisy image pairs appears to be more valuable than access to clean images. We also trained a supervised Noise Flow model and used samples generated from the model to train a DnCNN denoiser. We refer to this baseline as “DnCNN - NF synthesized”. The “DnCNN - NF synthesized” outperforms the “DnCNN-supervised” baseline which is consistent with the results reported in the Noise Flow paper [2]. However, it still significantly underperforms Noise2NoiseFlow.

Figure 4.2 shows qualitative denoising results from Noise2NoiseFlow and the aforementioned baselines. The results show that our model performs better in denoising, espe-

Model	PSNR	SSIM
Noise2Noise+DnCNN	51.57	0.977
DnCNN-supervised	51.32	0.980
DnCNN - NF synthesized	51.71	0.980
Noise2NoiseFlow(ours)	52.80	0.984

Table 4.3: Denoising results from a DnCNN trained with supervised noisy/clean paired data from SIDD-Medium, Noise2Noise with a DnCNN trained on noisy/noisy image pairs, a DnCNN trained with noise samples generated from a supervised Noise Flow model, and our denoiser model trained on the same noisy/noisy data measured by PSNR and SSIM on the test set.

cially in more severe situations (high ISO and low brightness). The estimated clean signal tends to be much smoother and cleaner for Noise2NoiseFlow than both of its baselines in terms of visual perception and PSNR in almost all cases. Taken together, our results demonstrate that the joint training of both an explicit noise model and a denoiser not only allows for weakly supervised training but also improves the resulting estimated denoiser.

4.4.3 Ablation Studies

We next investigate the design choices for our framework and their impact on the results. First, we conduct an ablation on the value of the weighting factor for the Noise2Noise loss. We explored a wide range of values, from $\alpha = 0$ to $\alpha = 2^{18}$. For each value, we computed the negative log-likelihood per dimension and the PSNR of the denoiser. The results are plotted in Fig. 4.3 and show that our results are relatively robust to the choice of α . While a value of $\alpha = 0$ produces reasonable results, better results are generally obtained with larger values of α . This indicates that the Noise2Noise loss in Eq. 3.6 plays an important role in stabilizing the training and ensuring consistency of the denoiser.

Next, we consider a different form of the loss function where we use the estimated

Figure 4.2: Denoising results from (b) DnCNN-supervised, (c) Noise2Noise trained with DnCNN, and (d) Noise2NoiseFlow on samples from SIDD-Medium testing data. The codes on the left indicate the ISO level as well as the lighting condition.

clean image based d_n for the noise model loss function with η . Formally, we use the

noise model objective

$$L_{\text{nm}}(\mathbf{t}_1; \mathbf{t}_2) = -\log p_{\mathbf{t}}(\mathbf{t}_1 | D(\mathbf{t}_1; \cdot)) - \log p_{\mathbf{t}}(\mathbf{t}_2 | D(\mathbf{t}_2; \cdot)); \quad (4.1)$$

instead of the one proposed in Equation 3.3, as discussed in Chapter 3. We refer to training based on this model as the self-sample loss, in comparison to the cross-sample loss. While a seemingly innocuous change, training based on Equation 4.1 becomes extremely unstable. In this case, the denoiser can converge to a degenerate solution of the identity function—namely, $D(\mathbf{t}) = \mathbf{t}$ —which allows the noise model to converge to a Dirac delta and the value $L_{\text{nm}}(\mathbf{t}_1; \mathbf{t}_2)$ goes to negative infinity. This behaviour can be alleviated with large values of β , which can be seen in Figure 4.3, where settings that resulted in diverged training are indicated with a cross at the value of the epoch before the divergence occurred. As the figure shows, values less than 2^{17} resulted in this behaviour. In contrast, the proposed loss function in Equation 3.3 is robust to the choice of β , even allowing training with a value of $\beta = 0$, which disables the \mathbf{e}_{dn} term from Equation 3.6 entirely. We also explored higher values for β (e.g., 2^{19}) but did not observe significant changes in behaviour.

We also explored different choices of denoiser architecture and noise model as our framework is agnostic to these specific choices. For the denoiser, beyond the DnCNN architecture, we also considered the U-Net [43] denoiser architecture used in [28]. For the noise model, beyond the Noise Flow-based model, we also considered the heteroscedastic Gaussian noise model, or noise level function (NLF), due to its ubiquity. We implemented the NLF as a variation on a Noise Flow architecture. Specifically, taking the signal-dependent and gain layers of the Noise Flow model, without any of the other low

Figure 4.3: Negative log-likelihood per dimension and PSNR results on test data as a function of the regularization term. Cross-sample loss refers to training with Eq. 3.3 and self-sample loss refers to training with Eq. 4.1. Cross marks indicate the loss at the last epoch in cases where training failed as evidenced by spikes in NLL and significant drops in PSNR. All experiments with the self-sample loss and 2^{16} ultimately diverge.

Denoiser	Noise model	NLL	D_{KL}	PSNR
DnCNN	Noise Flow	-3.501	0.0265	52.80
U-Net	Noise Flow	-3.500	0.0255	52.64
DnCNN	NLF	-3.461	0.0288	52.69
U-Net	NLF	-3.463	0.0332	52.13

Table 4.4: Performance on test data from our ablated models. Each row corresponds to a specific choice for the noise model and the denoiser architecture. NLF corresponds to our modified version of the heteroscedastic Gaussian model implemented as a bijective normalizing flow transformation.

layers, results in a model that is equivalent to the NLF.

The results of this experiment can be found in Table 4.4, which reports the negative log-likelihood per dimension, KL divergence, and PSNR of the resulting noise model and denoiser for all combinations of these choices. The results indicate that the choice of denoiser architecture is not particularly important. Both U-Net and DnCNN produce similar results to one another, for both choices of the noise model. However, we see that the use of the Noise Flow model over the heteroscedastic Gaussian noise model does provide a boost in performance for both noise modeling and denoising. Further, and consistent with results reported recently elsewhere [54], we see that a retrained heteroscedastic Gaussian noise model can outperform the parameters provided by camera manufacturers.

4.5 Training with Individual Noisy Images

Here we will evaluate the proposed R2RFlow variation described in Chapter 3. The model is trained according to the loss defined in equation 3.10. We use the same DnCNN architecture for D and the Noise Flow model for p , and report the results in Table 4.5, with this variation labelled as R2RFlow and compared against a clean-image supervised Noise

Model	NLL	D_{KL}	PSNR
Noise Flow	-3.502	0.0267	N/A
Noise2NoiseFlow	-3.501	0.0265	52.80
R2RFlow	-3.443	0.0983	50.08

Table 4.5: Performance on test data from our proposed R2RFlow formulation that requires only single noisy samples and no supervision in any forms compared to Noise2NoiseFlow (weak supervision) and the Noise Flow model (supervised).

Flow model and the noisy-pair weakly-supervised Noise2NoiseFlow. The results indicate that the R2RFlow approach yields a reasonable noise model, though significantly below the performance of Noise2NoiseFlow, particularly in terms of denoising. However, the experiment is enticing and suggests that this is a promising direction for future work.

4.6 Synthetic Noise Experiment

To demonstrate that our framework can retrieve the parameters of a supervised trained noise model, we have conducted a synthetic noise experiment. In this setting, we first trained a heteroscedastic Gaussian noise model, which was implemented as a flow layer in Noise Flow. For simplicity, we only took one camera and one ISO setting—namely, iPhone 7 and 800 as ISO level as we had adequate image data for training and evaluation. Under the mentioned setting, the model only has two trainable parameters—namely, μ and σ^2 . We then use this trained model to synthesize noisy image pairs for training a subsequent Noise2NoiseFlow model from scratch with only a heteroscedastic Gaussian layer as its noise model and DnCNN as its denoiser. The results shown in Figure 4.4 shows that our model can successfully retrieve the parameters of a trained NLF model.

Figure 4.4: Convergence curve of the two parameters α and β of the NLF model for a specific camera sensor and ISO level. NLF Parameter corresponds to the parameters learned by a supervised Noise Flow model and Reconstruction corresponds to the NLF parameters learned by a Noise2NoiseFlow model from synthetic data generated by the supervised Noise Flow model. As evidenced by the figures, the model can successfully retrieve the parameters.

4.7 Failure Cases

To investigate the model performance and spot bad performance, we aim to shed light on failure cases. Although no significant unrealistic behaviour was noticed, we visualize 5 noise samples with the word \mathbf{D}_{KL} for Noise2NoiseFlow in Figure 4.5. While the noise samples are not in the best alignment with the real samples, the generated noise patches do not look very unnatural.

Figure 4.5: Noise synthesis samples from (a) the AWGN model, (b) Camera NLF, (c) Calibrated P-G [54], (d) Noise Flow [2], and our proposed method, Noise2NoiseFlow, compared to the (f) real noise in SIDD for patches where Noise2NoiseFlow has the worst D_{KL} numbers.

Chapter 5

Conclusion

In this work, we introduced a novel framework for jointly training a noise model and a denoiser that does not require clean image data. Image noise modeling is a rich and long-studied problem in computer vision. Noise models evolved from simple models including Homoscedastic, and Heteroscedastic Gaussian models and Poisson-Gaussian formulations to more sophisticated models such as neural networks with many parameters.

With the recent development of deep learning methods, different DNN-based models were proposed that capture image noise in different setups. With Noise Flow as a successful example, we are now able to both compute the density estimation and generate synthesized noise for any given clean image and ISO setting. However, these models typically ignore the fact that gathering datasets of paired noisy-clean images is not trivial.

Estimating the unseen clean latent image is a challenging problem. The existing datasets have used either long exposure techniques or multiple capturing, both of which can be challenging and requires careful consideration of the scene as well as other captur-

ing factors, such as constant lighting, camera position, etc. These techniques may even be impossible in certain circumstances, such as scenes containing dynamic objects, and medical imaging domain where either multiple captures or long exposures can be dangerous for the subject.

The motivation set in motion numerous attempts in denoising to relax the clean image requirement. Noise2Noise [28] attempts to ease the problem by replacing the clean image target with another noisy observation. Their model trains a denoiser with minimal drops in denoising performance compared to their supervised baselines. Other attempts [25, 41, 4, 37, 5, 50, 35] Take a step further and remove the supervision entirely. However, the performance is not still quite on par with the supervised models.

This constraint has been relatively overlooked in the noise modeling domain. Existing noise models typically focus on improving the sample quality, or improving accuracy in estimating the parameters. In this work, we took a first cut at training a noise model without clean images. Rather, a formulation similar to that of Noise2Noise is adopted. Our model trains a noise model jointly with a denoiser using noisy-noisy image pairs. The evaluations reveal that the sample quality and the density function are on par with Noise Flow, the supervised counterpart of our model. This is a significant observation as it shows we can have almost the same performance without requiring clean image correspondence.

The evaluation on denoising also points out an interesting observation. The denoiser trained in our Noise2NoiseFlow joint training framework outperforms its supervised and the Noise2Noise baselines. This observation is important as it shows joint training framework has a positive impact on denoising performance as well. The denoising performance on the SIDD-evaluation shows an almost 1.2dB improvement in PSNR, which is signi -

cant. Additionally, the comparison against the “N2N+NF” shows the value that the joint training has on the noise modeling performance. When the models are trained separately, performance drop and some training instabilities are spotted.

We also touched upon the possibility of training a noise model with single noisy image pairs, by taking inspiration from the R2R [37] framework. The evaluations show a similar trend as denoising. While the model can still produce a reasonable model, the performance still lags behind its supervised and weakly-supervised baselines.

Also, the ablation studies on the denoiser and noise model architectures point out the generality of our framework. It is insensitive to the choice of denoiser and noise model as long as the noise model has a tractable density estimation function, and the whole function is differentiable. The combination of DnCNN, U-Net as a denoiser, with Noise Flow, and the NLF (implemented as a learned flow layer) shows that regardless of the choice for either the noise model or the denoiser, the generality of our model still holds.

Furthermore, we explored the possibility of training with self-sample loss formulation, which shows our cross-sample formulation gives the overall model more stability and is less sensitive to the value of α in the overall loss (Eq. 3.5). The model can still train even with no MSE in the overall loss term ($\alpha = 0$), which is a clear indication that the overall model is more robust and can handle the trade-off between L_{den} and L_{dn} better. The performance is also boosted compared to self-sample loss.

In sum, the Noise2NoiseFlow framework is tested under different scenarios and different aspects to make sure it learns a reliable noise model that can be used in downstream tasks. The synthetic noise experiment also indicates that Noise2NoiseFlow can learn the parameters learned by a supervised NLF model successfully. Our model produces an ad-

ditional denoiser at no cost. Usually, noise models are used in a second step of training a denoiser with synthetic noise. In our case, the denoiser is trained alongside the noise model.

5.1 Broader Impact

Noise models have several downstream use cases, which were listed in Chapter 2. Our work makes training a noise model easier and more practical for such tasks as it no longer requires clean image access. This is especially important in more critical tasks, such as medical imaging where having clean image references is extremely challenging, given that clean image estimation tools are dangerous in these domains. Although we did not investigate the medical imaging domain (e.g., Magnetic Resonance Imaging), our framework introduces such capability to perform well in those areas as well.

Furthermore, imaging in extreme low-light conditions such as astrophotography is another natural impact of our model. The gain factor plays an important role in such domains, which amplifies both the image and the noise, which makes image noise understanding more critical.

5.2 Future Work

Our work shows some potential for future directions. One natural direction is to remove the need for any source of supervision entirely. The preliminary experiments we did with the R2R [37] framework suggest that it may be indeed feasible, but it shows a promising direction for future work. The experiment with R2RFlow is a first cut at learning a

noise model with single noisy image measurements, and there is definitely room for more improvement in future works.

Another natural place to investigate for future improvement is the overall architectures of both the noise model and the denoiser. Although existing models in noise modeling and denoising show impressive results, they can still be improved by exploring other families of architectures, especially for the noise model.

Bibliography

- [1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S Brown. A high-quality denoising dataset for smartphone cameras. *CVPR*, 2018. ix, 1, 3, 16, 17, 29
- [2] Abdelrahman Abdelhamed, Marcus A Brubaker, and Michael S Brown. Noise flow: Noise modeling with conditional normalizing flows. *ICCV*, 2019. viii, x, 1, 2, 9, 10, 11, 25, 30, 31, 32, 34, 35, 44
- [3] Josue Anaya and Adrian Barbu. Renoir—a dataset for real low-light image noise reduction. *Journal of Visual Communication and Image Representation*, 51:144–154, 2018. 2, 3, 15, 16
- [4] Joshua Batson and Loic Royer. Noise2self: Blind denoising by self-supervision. In *ICML*, 2019. 3, 18, 25, 46
- [5] Sungmin Cha, Taeon Park, and Taesup Moon. Gan2gan: Generative noise learning for blind image denoising with single noisy image. *ICLR*, 2021. 5, 10, 19, 46
- [6] S Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *TIP*, 9(9):1532–1546, 2000. 1, 13

- [7] Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. *CVPR*, 2018. 5, 10, 19
- [8] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *TPAMI*, 16(8):2080–2095, 2007. 3, 13
- [9] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 10, 12
- [10] Alessandro Foi. Clipped noisy images: Heteroskedastic modeling and practical denoising. *Signal Processing*, 89(12):2609–2629, 2009. 1, 2, 9
- [11] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *TPAMI*, 17(10):1737–1754, 2008. 1, 9
- [12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS* 2014. 10
- [13] Shi Guo, Zifei Yan, Kai Zhang, Wangmeng Zuo, and Lei Zhang. Toward convolutional blind denoising of real photographs. *CVPR*, 2019. 13, 15
- [14] Glenn E Healey and Raghava Kondepudy. Radiometric ccd camera calibration and noise estimation. *TPAMI*, 16(3):267–276, 1994. 1

- [15] Bernardo Henz, Eduardo SL Gastal, and Manuel M Oliveira. Synthesizing camera noise using generative adversarial networks. *IEEE Transactions on Visualization and Computer Graphics*, 27(3):2123–2135, 2020. 1, 10
- [16] Zhiwei Hong, Xiaocheng Fan, Tao Jiang, and Jianxing Feng. End-to-end unpaired image denoising with conditional adversarial networks. *AAAI*, 2020. 5, 10
- [17] Wei Hu, Lechao Xiao, and Jeffrey Pennington. Provable benefit of orthogonal initialization in optimizing deep linear networks. *arXiv preprint arXiv:2001.05992*, 2020. 35
- [18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015. viii, 13
- [19] Geonwoon Jang, Wooseok Lee, Sanghyun Son, and Kyoung Mu Lee. C2n: Practical generative noise modeling for real-world denoising. *ICCV*, 2021. 5, 10
- [20] Dong-Wook Kim, Jae Ryun Chung, and Seung-Won Jung. Grdn: Grouped residual dense network for real image denoising and gan-based real-world noise modeling. In *CVPR 2019*. 5, 10
- [21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 24, 31
- [22] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS 2018*. 12
- [23] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *TPAMI*, 2020. 10

- [24] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *TPAMI*, 43(11):3964–3979, 2021. 2
- [25] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. *CVPR*, 2019. 3, 18, 25, 46
- [26] Darwin T. Kuan, Alexander A. Sawchuk, Timothy C. Strand, and Pierre Chavel. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7(2):165–177, 1985. doi: 10.1109/TPAMI.1985.4767641. 13
- [27] Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. *NeurIPS*, 2019. 18
- [28] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. In *ICML*, 2018. 3, 18, 22, 32, 34, 38, 46
- [29] Ce Liu, Richard Szeliski, Sing Bing Kang, C Lawrence Zitnick, and William T Freeman. Automatic estimation and removal of noise from a single image. *TPAMI*, 30(2):299–314, 2007. 1, 13
- [30] Xinhao Liu, Masayuki Tanaka, and Masatoshi Okutomi. Practical signal-dependent noise parameter estimation from a single noisy image. *TIP*, 23(10):4361–4371, 2014. 1, 8
- [31] Yang Liu, Saeed Anwar, Zhenyue Qin, Pan Ji, Sabrina Caldwell, and Tom Gedeon.

Disentangling noise from images: A low-based image denoising neural network.
arXiv preprint arXiv:2105.04746, 2021. 1

- [32] Markku Makitalo and Alessandro Foi. Optimal inversion of the generalized anscombe transformation for poisson-gaussian noise. *IEEE transactions on image processing* 22(1):91–103, 2012. 9
- [33] Christopher A Metzler, Ali Mousavi, Reinhard Heckel, and Richard G Baraniuk. Unsupervised learning with stein's unbiased risk estimator. arXiv preprint arXiv:1805.10531, 2018. 19
- [34] Amr M Mohsen, Michael F Tompsett, and Carlo H Sèquin. Noise measurements in charge-coupled devices. *IEEE Transactions on Electron Devices* 49:75. 8
- [35] Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. *CVPR*, 2020. 5, 19, 25, 46
- [36] Seonghyeon Nam, Youngbae Hwang, Yasuyuki Matsushita, and Seon Joo Kim. A holistic approach to cross-channel image noise modeling and its application to image denoising. In *CVPR*, 2016. 2
- [37] Tongyao Pang, Huan Zheng, Yuhui Quan, and Hui Ji. Recorrputed-to-recorrputed: Unsupervised deep learning for image denoising. *CVPR*, 2021. 5, 6, 19, 25, 46, 47, 48
- [38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Py-

torch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.

31

- [39] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In *CVPR*, 2017. ix, 3, 15, 16, 17
- [40] Javier Portilla, Vasily Strela, Martin J Wainwright, and Eero P Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *TIP*, 12(11): 1338–1351, 2003. 1, 13
- [41] Yuhui Quan, Mingqin Chen, Tongyao Pang, and Hui Ji. Self2self with dropout: Learning self-supervised denoising from single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1890–1898, 2020. 19, 46
- [42] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015. 10
- [43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015. ix, 13, 14, 38
- [44] Tamara Seybold, Christian Keimel, Marion Knopp, and Walter Stechele. Towards an evaluation of denoising algorithms with respect to realistic camera noise. In *2013 IEEE International Symposium on Multimedia*. IEEE, 2013. 2, 15
- [45] Shakarim Soltanayev and Se Young Chun. Training deep learning based denoisers without ground truth data. *NeurIPS*, 2018. 19

- [46] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *CVPR*, 2018. 19
- [47] Wei Wang, Xin Chen, Cheng Yang, Xiang Li, Xuemei Hu, and Tao Yue. Enhancing low light videos by exploring high sensitivity camera noise. In *ICCV*, 2019. 2
- [48] Valentin Wolf, Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Deflow: Learning complex image degradations from unpaired data with conditional flows. In *CVPR*, 2021. 10
- [49] Jun Xu, Hui Li, Zhetong Liang, David Zhang, and Lei Zhang. Real-world noisy image denoising: A new benchmark. *arXiv preprint arXiv:1804.02603*, 2018. 3, 17
- [50] Jun Xu, Yuan Huang, Ming-Ming Cheng, Li Liu, Fan Zhu, Zhou Xu, and Ling Shao. Noisy-as-clean: learning self-supervised denoising from corrupted image. *TIP*, 29: 9316–9329, 2020. 5, 19, 25, 46
- [51] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *TIP*, 26(7): 3142–3155, 2017. viii, 6, 13, 25
- [52] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *TIP*, 27(9):4608–4622, 2018. 13
- [53] Ming Zhang and Bahadir K Gunturk. Multiresolution bilateral filtering for image denoising. *TIP*, 17(12):2324–2333, 2008. 13
- [54] Yi Zhang, Hongwei Qin, Xiaogang Wang, and Hongsheng Li. Rethinking noise synthesis and modeling in raw denoising. In *ICCV*, 2021. x, 2, 9, 32, 34, 40, 44

- [55] Yuqian Zhou, Jianbo Jiao, Haibin Huang, Yang Wang, Jue Wang, Honghui Shi, and Thomas Huang. When awgn-based denoiser meets real noises. In *AAAI*, 2020. 2, 15
- [56] Fengyuan Zhu, Guangyong Chen, and Pheng-Ann Heng. From noise modeling to blind image denoising. In *CVPR*, 2016. 10